



European Space Astronomy Centre
P.O. Box 78
28691 Villanueva de la Cañada
Madrid
Spain ☐
T +34 91 8131 100
F +34 91 8131 139
www.esa.int ☐

DOCUMENT

Cluster Science Archive - Software Validation Specification

Prepared by	Science Archives Team
Reference	SAT_CSA_TD_1.0_01_SVS_07Oct2013
Issue	1
Revision	0
Date of Issue	07/10/2013
Status	Draft
Document Type	SVVP
Distribution	SAT team (SRE-OE)



APPROVAL

Title CSA - Software Validation Specification	
Issue 1	Revision 0
Author Science Archives Team	Date 07/10/2013
Approved by	Date
Head of the ESAC Science Archives Team	TBD

CHANGE LOG

Reason for change	Issue	Revision	Date

CHANGE RECORD

Issue 1	Revision 0		
Reason for change	Date	Pages	Paragraph(s)

Table of contents:

Contents

1	INTRODUCTION	8
2	APPLICABLE AND REFERENCE DOCUMENTS	8
3	TERMS, DEFINITIONS AND ABBREVIATED TERMS	9
4	SOFTWARE OVERVIEW	10
5	SOFTWARE VALIDATION SPECIFICATION TASK IDENTIFICATION.....	11
5.1	Task and criteria.....	11
5.2	Features to be tested	11
5.3	Features not to be tested.....	12
5.4	Test pass-fail criteria.....	12
5.5	Items that cannot be validated by test	12
5.6	Environment	13
5.6.1	User Interface	14
5.6.2	CAIO	14
5.6.3	Server Machine	15
5.6.4	Database.....	15
6	SOFTWARE VALIDATION TESTING SPECIFICATION DESIGN	16
6.1	General	16
6.2	Test designs	16
6.2.1	Test design IMPORT-BASIC	16
6.2.2	Test design IMPORT-LOGGING.....	17
6.2.3	Test design INGEST-BASIC	18
6.2.4	Test design INGEST-LOGGING.....	18
6.2.5	Test design DD-BASIC	19
6.2.6	Test design DD-RETRIEVAL	20
6.2.7	Test design DD-LOGGING	20
6.2.8	Test design GRAPHICS-BASIC.....	21
6.2.9	Test design GRAPHICS-LOGGING	21
6.2.10	Test design UI-BASIC.....	22
6.2.11	Test design UI-SEARCH.....	23
6.2.12	Test design UI-RESULTS	24
6.2.13	Test design UI-RETRIEVAL.....	25
6.2.14	Test design UI-PLOTTING.....	26
6.2.15	Test design UI-SECURITY	27
6.2.16	Test design UI-LOGGING	28
6.2.17	Test design UI-PERFORMANCE.....	29
6.2.18	Test design UI-SCALABILITY.....	30
6.2.19	Test design UI-STRESS	31
6.2.20	Test design CAIO-BASIC	32
6.2.21	Test design CAIO-SEARCH.....	33
6.2.22	Test design CAIO-RESULTS	34
6.2.23	Test design CAIO-RETRIEVAL.....	35
6.2.24	Test design CAIO-LOGGING	36
6.2.25	Test design CAIO-SECURITY	37
7	SOFTWARE VALIDATION TEST SPECIFICATION	37
7.1	General	37
7.2	IMPORT-BASIC Test Cases	38
7.2.1	Test Case IMPORT-BASIC-01: Start-up of Import server	38

7.2.2	Test Case IMPORT-BASIC-02: Import directories available	39
7.2.3	Test Case IMPORT-BASIC-03: Ingestion Request File validity	39
7.2.4	Test Case IMPORT-BASIC-04: Sorting of Ingestion Request Files	40
7.2.5	Test Case IMPORT-BASIC-05: Ingestion Request File format	41
7.2.6	Test Case IMPORT-BASIC-06: Import Data files	41
7.2.7	Test Case IMPORT-BASIC-07: Import Header files	42
7.2.8	Test Case IMPORT-BASIC-08: Import Metadata files	42
7.2.9	Test Case IMPORT-BASIC-09: Ingest queue	43
7.3	IMPORT-LOGGING Test Cases	44
7.3.1	Test case IMPORT-LOGGING-01: Log output	44
7.3.2	Test case IMPORT-LOGGING-02: Report Output	44
7.4	INGEST-BASIC Test Cases	45
7.4.1	Test Case INGEST-BASIC-01: Start-up of Ingest server	45
7.4.2	Test Case INGEST-BASIC-02: Daemon process	45
7.4.3	Test Case INGEST-BASIC-03: Associated import logs	46
7.4.4	Test Case INGEST-BASIC-04: Ingest Data files	47
7.4.5	Test Case INGEST-BASIC-05: Ingest XML metadata files	47
7.4.6	Test Case INGEST-BASIC-06: Ingest Header files	48
7.4.7	Test Case INGEST-BASIC-07: Ingest Inventory files	49
7.4.8	Test Case INGEST-BASIC-08: Ingest Graphical Metadata CSV files	49
7.5	INGEST-LOGGING Test Cases	50
7.5.1	Test case INGEST-LOGGING-01: Log output	50
7.5.2	Test case INGEST-LOGGING-02: Report Output for Ingestion Request Files sent by CAA	50
7.5.3	Test case INGEST-LOGGING-03: Report Output for Ingestion Request Files generated at CSA	51
7.6	DD-BASIC Test Cases	52
7.6.1	Test Case DD-BASIC-01: Start-up of Data Distribution service	52
7.6.2	Test Case DD-BASIC-02: Port open for Data Distribution service	52
7.7	DD-RETRIEVAL Test Cases	53
7.7.1	Test Case DD-RETRIEVAL-01: Direct download service	53
7.7.2	Test Case DD-RETRIEVAL-02: Asynchronous download service	53
7.8	DD-LOGGING Test Cases	54
7.8.1	Test case DD-LOGGING-01: Log output	54
7.8.2	Test case DD-LOGGING-02: Logging levels	55
7.9	GRAPHICS-BASIC Test Cases	55
7.9.1	Test Case GRAPHIC-BASIC-01: Generate On-demand plot	55
7.9.2	Test Case GRAPHIC-BASIC-02: Generate pre-generated plot	56
7.10	GRAPHICS-LOGGING Test Cases	56
7.10.1	Test case GRAPHICS-LOGGING-01: Log output	56
7.10.2	Test case GRAPHICS-LOGGING-02: Logging levels	57
7.11	UI-BASIC Test Cases	58
7.11.1	Test Case UI-BASIC-01: Application start-up using Linux OS	58
7.11.2	Test Case UI-BASIC-02: Application start-up using Windows OS	58
7.11.3	Test Case UI-BASIC-03: Application start-up using Mac OS	59
7.11.4	Test Case UI-BASIC-04: Start-up from outside ESAC firewall	59
7.11.5	Test Case UI-BASIC-05: Menu and Toolbar	60
7.11.6	Test Case UI-BASIC-06: Application exit	61
7.12	UI-SEARCH Test Cases	61
7.12.1	Test case UI-SEARCH-01: No data returned notification	61
7.12.2	Test case UI-SEARCH-02: Search for all datasets	62
7.12.3	Test case UI-SEARCH-03: Cancel search	62
7.12.4	Test case UI-SEARCH-04: Search for all inventory data	63
7.12.5	Test case UI-SEARCH-05: Search datasets with a given criteria	63
7.12.6	Test case UI-SEARCH-06: Clear search	64

7.12.7	Test case UI-SEARCH-07: Search inventory data with a given criteria	65
7.12.8	Test case UI-SEARCH-08: Validation of input fields at Search	65
7.13	UI-RESULTS Test Cases	66
7.13.1	Test case UI-RESULTS-01: Short-list of datasets	66
7.13.2	Test case UI-RESULTS-02: Display Results Details	67
7.13.3	Test case UI-RESULTS-03: Results pagination	68
7.13.4	Test case UI-RESULTS-04: Results dynamic page size	69
7.13.5	Test case UI-RESULTS-05: Change Time constraint at Data Request tab	70
7.13.6	Test case UI-RESULTS-06: Selection of dataset Units	70
7.13.7	Test case UI-RESULTS-07: Clear datasets selection	71
7.13.8	Test case UI-RESULTS-08: Save datasets in a profile	72
7.13.9	Test case UI-RESULTS-09: Choose delivery format	72
7.13.10	Test case UI-RESULTS-10: Add results to Shopping Basket	73
7.13.11	Test case UI-RESULTS-11: Remove results from Shopping Basket	74
7.13.12	Test case UI-RESULTS-12: Docking/Undocking Details windows	75
7.13.13	Test case UI-RESULTS-13: Docking/Undocking Results tabs	75
7.14	UI-RETRIEVAL Test Cases	76
7.14.1	Test case UI-RETRIEVAL-01: Direct Download of Cluster files	76
7.14.2	Test case UI-RETRIEVAL-02: Direct Download of non-Cluster files	77
7.14.3	Test case UI-RETRIEVAL-03: Download of Cluster files from Shopping Basket	78
7.14.4	Test case UI-RETRIEVAL-04: Download of non-Cluster files from Shopping Basket	78
7.15	UI-PLOTTING Test Cases	79
7.15.1	Test case UI-PLOTTING-01: Selection of time interval	79
7.15.2	Test case UI-PLOTTING-02: Selection of Plot Type	80
7.15.3	Test case UI-PLOTTING-03: Selection of Graphical Products and sorting	81
7.15.4	Test case UI-PLOTTING-04: Display Plot Result	81
7.15.5	Test case UI-PLOTTING-05: Save Graphical products in a Graphical profile	82
7.15.6	Test case UI-PLOTTING-06: Plot with no data returned	83
7.15.7	Test case UI-PLOTTING-07: Stack of plots and navigation	84
7.15.8	Test case UI-PLOTTING-08: Plots next/previous duration	85
7.15.9	Test case UI-PLOTTING-09: Docking/Undocking Plots windows	86
7.15.10	Test case UI-PLOTTING-10: Retrieve plot in Postscript format	86
7.15.11	Test case UI-PLOTTING-11: Retrieve CEF source files	87
7.15.12	Test case UI-PLOTTING-12: Retrieve plot file in PNG format	88
7.16	UI-SECURITY Test Cases	88
7.16.1	Test case UI-SECURITY-01: Log in/ logout	88
7.16.2	Test case UI-SECURITY-02: Register as a New User	89
7.16.3	Test case UI-SECURITY-03: Access to public services	90
7.16.4	Test case UI-SECURITY-04: Access to non-public services	90
7.17	UI-LOGGING Test Cases	91
7.17.1	Test case UI-LOGGING-01: Log Console	91
7.17.2	Test case UI-LOGGING-02: Logging levels	92
7.18	UI-PERFORMANCE Test Cases	92
7.18.1	Test case UI-PERFORMANCE-01: Web interface initialisation	93
7.18.2	Test case UI-PERFORMANCE-05: Retrieval of CEF product from Shopping-Basket	94
7.18.3	Test case UI-PERFORMANCE-06: Retrieval of non-CEF product from Shopping-Basket	94
7.18.4	Test case UI-PERFORMANCE-09: Availability of requested files	95
7.19	UI-SCALABILITY Test Cases	96
7.19.1	Test case UI-SCALABILITY-06: Scalability of direct download retrieval close to maximum size allowed	97
7.19.2	Test case UI-SCALABILITY-07: Scalability of Shopping Basket retrieval close to maximum size allowed	97
7.19.3	Test case UI-SCALABILITY-09: Scalability of typical workload scenario for a short time range	98
7.19.4	Test case UI-SCALABILITY-10: Scalability of typical workload scenario for a large time range	98
7.20	UI-STRESS Test Cases	99

7.20.1	Test case UI-STRESS-01: Stress Server Layer with a typical workload scenario	99
7.20.2	Test case UI-STRESS-02: Recoverability of the system	100
7.21	CAIO-BASIC Tests Cases	100
7.21.1	Test Case CAIO-BASIC-01: Web pages load and display in common browsers on Linux OS	100
7.21.2	Test Case CAIO-BASIC-02: Web pages load and display in common browsers on Windows OS	101
7.21.3	Test Case CAIO-BASIC-03: Web pages load and display in common browsers on Mac OS	102
7.21.4	Test Case CAIO-BASIC-04: Web pages load and display outside the ESAC firewall	102
7.21.5	Test Case CAIO-BASIC-05: CAIO menu bar functionality	103
7.21.6	Test Case CAIO-BASIC-06: CAIO links	103
7.21.7	Test Case CAIO-BASIC-07: Login requested.....	104
7.21.8	Test Case CAIO-BASIC-08: AIO client on Linux OS.....	105
7.21.9	Test Case CAIO-BASIC-09: AIO client on Windows OS.....	105
7.21.10	Test Case CAIO-BASIC-10: AIO client on Mac OS.....	106
7.21.11	Test Case CAIO-BASIC-11: Authentication via 'wget'	106
7.22	CAIO-SEARCH Test Cases.....	107
7.22.1	Test case CAIO-SEARCH-01: No data returned notification.....	107
7.22.2	Test case CAIO-SEARCH-02: No metadata returned notification.....	108
7.22.3	Test case CAIO-SEARCH-03: Metadata queries	108
7.22.4	Test case CAIO-SEARCH-04: Product queries	109
7.22.5	Test case CAIO-SEARCH-05: Metadata queries Return Type	109
7.22.6	Test case CAIO-SEARCH-06: Metadata queries constrained by PAGE_SIZE and PAGE.....	110
7.22.7	Test case CAIO-SEARCH-07: Complex metadata queries with QUERY parameter	111
7.22.8	Test case CAIO-SEARCH-08: Sorting metadata queries.....	111
7.22.9	Test case CAIO-SEARCH-09: Header Requests	112
7.22.10	Test case CAIO-SEARCH-10: Wrong query format	112
7.23	CAIO-RESULTS Test Cases	113
7.23.1	Test case CAIO-RESULTS-01: Results in VOTable.....	113
7.23.2	Test case CAIO-RESULTS-02: Results in JSON	114
7.23.3	Test case CAIO-RESULTS-03: Results in CSV.....	114
7.24	CAIO-RETRIEVAL Test Cases	115
7.24.1	Test case CAIO-RETRIEVAL-01: Synchronous retrieve of data product.....	115
7.24.2	Test case CAIO-RETRIEVAL-02: Retrieve of header files.....	116
7.24.3	Test case CAIO-RETRIEVAL-03: Retrieve of data products from multiple datasets.....	116
7.24.4	Test case CAIO-RETRIEVAL-04: Retrieve of Inventory data	117
7.24.5	Test case CAIO-RETRIEVAL-05: Asynchronous retrieval of data product	118
7.24.6	Test case CAIO-RETRIEVAL-06: Asynchronous retrieval of data product from multiple datasets	118
7.24.7	Test case CAIO-RETRIEVAL-07: CAIO user manual	119
7.24.8	Test case CAIO-RETRIEVAL-08: AIO command line client.....	120
7.24.9	Test case CAIO-RETRIEVAL-09: Retrieve metadata files	120
7.25	CAIO-LOGGING Test Cases	121
7.25.1	Test case CAIO-LOGGING-01: Log output.....	121
7.25.2	Test case CAIO-LOGGING-02: Logging levels.....	121
7.26	CAIO-SECURITY Test Cases	122
7.26.1	Test case CAIO-SECURITY-01: Log in/ logout	122
7.26.2	Test case CAIO-SECURITY-02: Access to public services	123
7.26.3	Test case CAIO-SECURITY-03: Access to non-public services	123
8	SOFTWARE VALIDATION TEST PROCEDURES	124
9	SOFTWARE VALIDATION ANALYSIS, INSPECTION, REVIEW OF DESIGN	124
10	VALIDATION TEST PLATFORMS REQUIREMENTS	125
10.1	Performance, scalability and stress testing considerations.....	125
10.2	The GRINDER.....	125
11	SOFTWARE VALIDATION SPECIFICATION ADDITIONAL INFORMATION.....	127
11.1	Requirement to test/analysis/inspection/review of design traceability matrix.....	127



11.2 Test/analysis/inspection/review of design to requirement traceability matrix.....130

11.3 Requirement to Test Design traceability matrix.....130



1 INTRODUCTION

The purpose of this document is to describe the system testing, analysis, inspection and review of design specifications to be performed as part of the software validation plan for the Cluster Science Archive (CSA).

System tests verify that the software system meets the software requirements with primary purpose to fully exercise the complete system. If possible, tests must be executed by an independent team in a development and/or operational environment. Knowledge of the system is not needed, so system testing is essentially a black-box test.

Test cases described in this document are mainly based on functional requirements. Non-functional requirements (e.g. interface, resource, reliability and safety tests) are out of the scope of the document with the exception of the performance, scalability and security tests, which have proven to be of key importance for any release of a science archive system.

This document is fully compatible with the European Cooperation for Space Standardization (ECSS) (see [ECSS-E-ST-40C]), and it constitutes the Software Validation Specification (SVS) deliverable, which aims to validate process implementation and activities with respect to the technical specification and the requirements baseline.

2 APPLICABLE AND REFERENCE DOCUMENTS

[ABSI]	<i>Archives Building System Infrastructure Developers Guide, 5 Nov 2009</i>
[ECSS-E-ST-40C]	<i>ECSS Space Engineering Software, ECSS, 6 March 2009</i>
[SPMP]	<i>Software Project Management Plan, SAT, 30 May 2011</i>
[CFA_EST_UR]	<i>User Requirements for the Cluster Final Archive, 28 April 2011 (v 1.0)</i>
[CFA_URD_Implementation_Table]	<i>CFA Implementation Matrix, 6 June 2013</i>
[CAA-CFA-ICD]	<i>CAA-CFA Interface Control Document, 21 May 2013 (v 0.5)</i>
[CAA_GRAPHICS]	<i>CAA – Graphical Panels, 12 Aug 2013 (v 2.3)</i>
[CAA_PANEL_XML]	<i>CAA/CFA Panel XML – description of datasets, 7 Jul 2013 (v 0.6)</i>
[CSA_URD_Test_Matrix]	<i>SAT_CSA_URD_Tests_Matrix_17Oct2013.xlsx (Traceability matrix between URD and Test designs-Test Cases)</i>

3 TERMS, DEFINITIONS AND ABBREVIATED TERMS

ABSI	Archives Building Software Infrastructure
CAA	Cluster Active Archive
CAIO	CSA Archive Inter-Operability subsystem
CFA	Cluster Final Archive (original name of CSA)
CI	Configuration Item
CL	Client Layer
CSA	Cluster Science Archive
CSA/DT	Cluster Science Archive Development Team
CSA/KP	Cluster Science Archive Key Person
CSV	Comma Separated Value format
DD	Cluster science archive Data Distribution subsystem
DL	Data Layer
H/SAT	Head of SAT
ICD	Interface Control Document
IRF	Ingestion Request File
QM	Cluster science archive Query Manager subsystem
RM	Cluster science archive Request Manager subsystem
SAT	Science Archives Team
SL	Server Layer
SVR	Software Validation Report
SVS	Software Validation Specification
UI	Cluster science archive User Interface subsystem
UR	User Requirement
URD	User Requirement Document

The tests to be performed during the validation process will fall into one of the following validation methods:

- **Test (T)**
This method involves the execution of a test procedure so that the output obtained shows the compliance to one or several requirements.
- **Analysis (A)**
Some requirements may need validation by analysis. This could be in the case of availability or performance requirements where the time period used in the requirement may be impossible to validate in a single test and extrapolation is necessary. This method will usually be combined with the Test method.
- **Inspection (I)**
This method is used to validate those requirements where a test or analysis is not applicable, which can only be validated by a visual inspection.
- **Review of Design (D)**
This method is applicable to validate the requirements that cannot be validated following any of the methods mentioned above. In other words, the system design

covering such requirements implicitly demonstrates the compliance against the specification.

4 SOFTWARE OVERVIEW

The CSA relies on the global SAT infrastructure for archives building purposes called ABSI, so a brief discussion of the main software systems under test can be found in the architectural description document [ABSI].

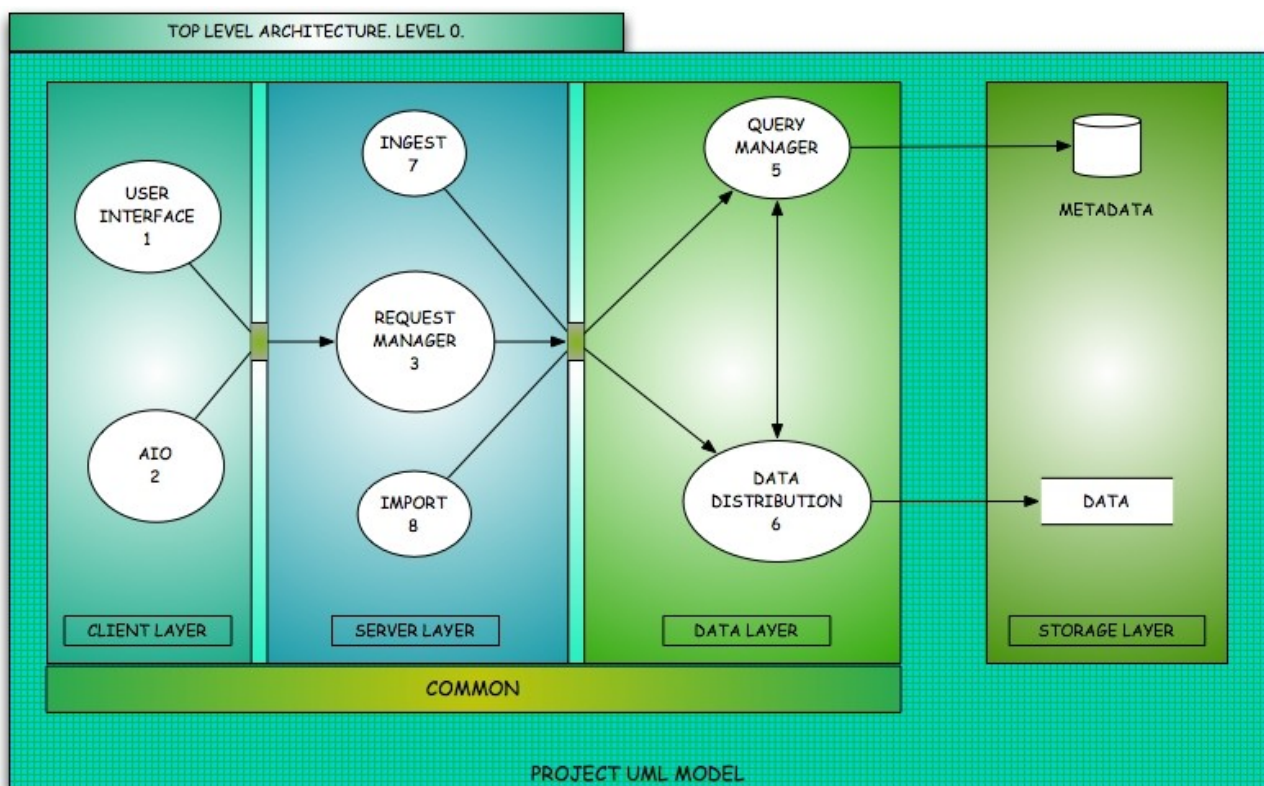


Figure 1: ABSI Top Level Architecture

Following the ABSI architecture (see Figure 1), these are the software configuration items applicable for the Cluster Science Archive:

Configuration Item	Layer	Function
QM	DL	Query Manager
DD	DL	Data Distribution
GRAPHICS	DL	Graphical Plotting subsystem
IMPORT	SL	Data import into file system
INGEST	SL	Metadata ingestion into database



RM	SL	Request Manager
UI	CL	Graphical User Interface
CAIO	CL	Inter-operability

Table 1: CSA configuration items

In addition, a detailed description of its external interface can be found on the CAA-CFA interface control document [CAA-CFA-ICD].

5 SOFTWARE VALIDATION SPECIFICATION TASK IDENTIFICATION

The approach to be utilized for the software validation specification is detailed below.

5.1 Task and criteria

The CSA/DT checks the validation of the CSA software deliverables. If during a task review problems are discovered concerning the correct conclusion of the task, a decision should be made also concerning the iteration of the task. Guidelines for the following cases are:

- The CSA/DT is unable to complete a task, for example because of lack of knowledge or manpower. In this case, it is the responsibility of the CSA/DT to report the problem to the CSA/KP, who will try to solve the problem and make sure the task is completed as described in the [SPMP]. However, if the CSA/KP is unable to do so, problem will be scaled up to the H/SAT.
- A structure error is found on the task, for example that the output does not comply with the requirements. In this case, the CSA/DT is responsible to perform the task again to reproduce the error. If error persists, the suspension and resumption criteria detailed in section 5.4 will apply. If necessary the CSA/KP or the H/SAT allocates time for the CSA/DT to fix the error found following the SAT software development life-cycles described in the [SPMP].
- An item was forgotten during the execution of the task. Depending on the severity of the item, the CSA/KP decides whether the entire task or only a part of the task needs to be redone or that no action is taken. In other cases the CSA/KP will decide which actions are to be taken, consulting the H/SAT if necessary.

5.2 Features to be tested

The features to be tested involve all CIs and are based on the requirements baseline which comprises the following areas (functional and non-functional):

Functional/Non-Functional Area	Identifier
Authentication and Authorization	SECURITY
Basic functionality	BASIC
Data retrieval	RETRIEVAL
Logging for analytical statistics	LOGGING
Metadata search	SEARCH
Metadata search results	RESULTS
Plot of graphical products	PLOTTING
Performance	PERFORMANCE
Scalability	SCALABILITY
Stress testing	STRESS

Table 2: Functional and non-functional areas of test cases

5.3 Features not to be tested

Specific test will not be provided for the RM neither the QM configuration items, but they will be indirectly tested by other CIs (DD, IMPORT, INGEST and UI).

5.4 Test pass-fail criteria

The general criteria to determine whether or not tests are passed will be defined at the test case specification (section 7). The criterion to suspend and/or resume the testing follows below:

Suspension criteria:

- Unavailability of external dependency during execution.
- Defect introduced not allowing further testing (i.e. a blocker bug)¹
- Critical path deadline is missed, so that the client will not accept delivery even if all testing is completed.
- Lack of testing resources.

Resumption requirements:

- External dependent systems become available again.
- Fix successfully implemented and testing team is notified to continue testing.
- Deadline extended.
- Testing resources become available.

5.5 Items that cannot be validated by test

There will be tasks and items under validation that cannot be validated by test. These will be justified in a case-by-case basis within the software validation report. For completeness, please refer to the method of test – requirements traceability matrix at chapter 0.

¹ Major or critical defects may not constitute suspension if other areas could continue to be tested.

5.6 Environment

This section details the computing environment in which the software validation will take place. Any environment specific to a given test case will be described in the description of the test case itself.

Three versions of the CSA are available (internal, development and operational). Each environment has its own machine (virtual or physical) so differences among their specifications and capabilities must be taken into account when planning the tests if done on different environments.

The internal environment is used for internal unit and integration tests within the SAT, whereas the development environment is used for beta releases. Hence, the most suitable environment to perform software validation tests is the development one (see Figure 2). Nonetheless, performance, scalability and stress tests might be run on the operational environment in order to test the real response times obtained.

Target version will be clearly defined for every test before the start of the test. Also references to database and repository used will be provided.

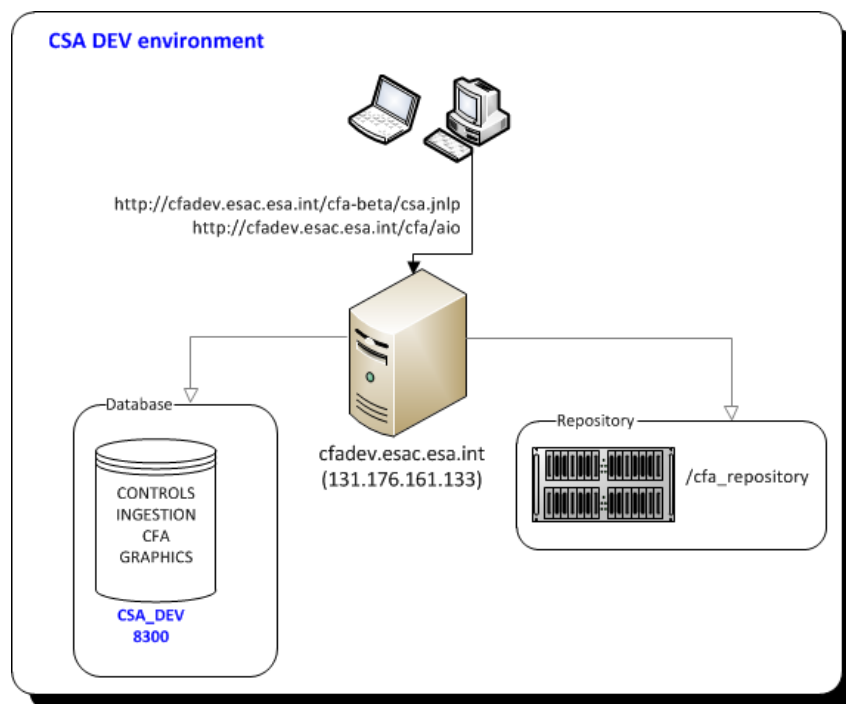


Figure 2: CSA development environment

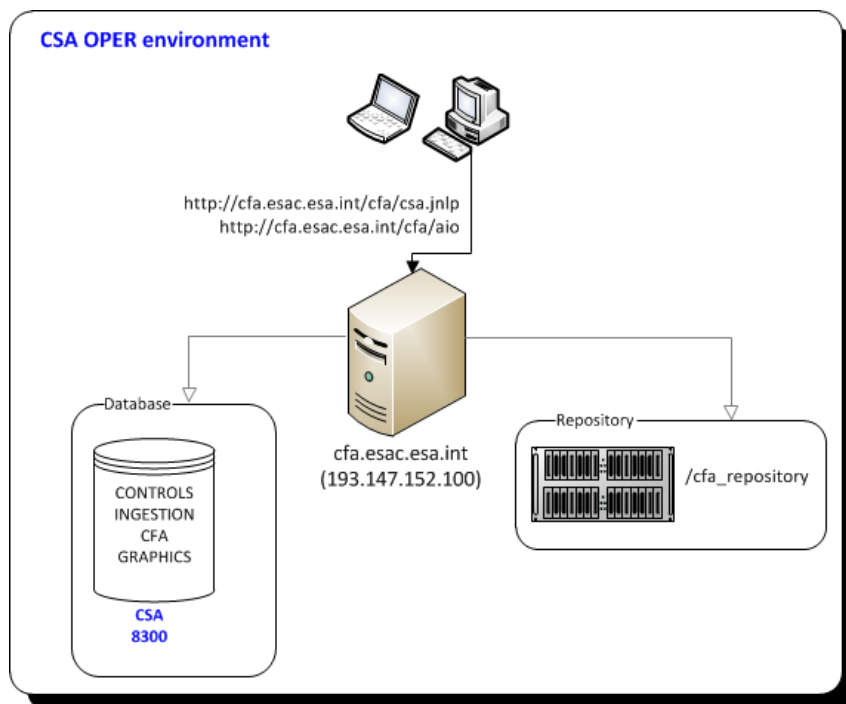


Figure 3: CSA operational environment

5.6.1 User Interface

The CSA user interface is a Java desktop application accessible through the following web pages:

- <http://cfa.esac.esa.int/cfa/csa.jnlp> for the operational version
- <http://cfadev.esac.esa.int/cfa-beta/csa.jnlp> for the beta version
- <http://cfaint.esac.esa.int/cfa-int/csa.jnlp> for the internal version

The user interface uses Spring Remoting for Java serialization via HTTP to interact with the server.

5.6.2 CAIO

The Cluster Science Archive Inter-Operability subsystem (CAIO) is a system that allows users to have direct access to the contents of the CSA without invoking the CSA user interface java application. The CAIO uses the same back-end as the user interface and makes requests for data and metadata via HTTP requests.

Queries to the CAIO are made through the following URLs:

- <http://cfa.esac.esa.int/cfa/aio> for the operational version
- <http://cfadev.esac.esa.int/cfa/aio> for the beta version
- <http://cfaint.esac.esa.int/cfa/aio> for the internal version

5.6.3 *Server Machine*

The CSA server is composed of three modules:

- **Server Layer:** It is a Java application in a Jakarta Tomcat server. It listens to port 80 and serves the requests of the user interface.
- **Data Distribution:** It is a stand-alone Java application. It serves requests for files (including concatenation).
- **Query Manager:** It is a stand-alone Java application. It serves request for data from the database.

The Server Layer receives requests from the user interface and distributes them either to the data distribution or the query manager, and replies back to the user interface. All internal communication is done through Java RMI (Remote Method Invocation).

The CSA server is located at:

- **cfa.esac.esa.int:** for the operational archive
- **cfadev.esac.esa.int:** for the beta archive
- **cfaint.esac.esa.int:** for the internal archive

In addition, ftp servers to retrieve the archive files are installed and running. In order to avoid firewall and similar network problems (ports closed, restrictive policies,...), they are accessible through Tomcat servers:

- http://cfa.esac.esa.int/ftp_public/ for the operational archive
- http://cfadev.esac.esa.int/ftp_public for the beta archive
- http://cfaint.esac.esa.int/ftp_public for the internal archive

5.6.4 *Database*

The CSA connects to a Postgresql database server. Inside the server, the database where the archive data is stored is called 'cfa'.

There is one database server per environment located in the corresponding machine and the port it uses in every environment is 8300.

6 SOFTWARE VALIDATION TESTING SPECIFICATION DESIGN

6.1 General

The specification of the testing design that follows is grouped by CIs (section 4, Table 1) and functional/non-functional areas (section 5.2, Table 2). The naming convention for the test design is the CI acronym followed by the area identifier.

Note that no traceability information is directly provided within this document, please refer to the traceability matrix at [CSA_URD_Test_Matrix] instead. That document traces every User Requirement to the correspondent Test Designs and Test Cases (see section 7). The traceability matrix to Test Cases has been split into several sheets for readability.

6.2 Test designs

6.2.1 Test design *IMPORT-BASIC*

6.2.1.1 General

This test design establishes procedures for the validation on the basic functionality of the import software. Being a largely automatic process, the success or otherwise of these tests can be determined from the output logs (verified in section 6.2.2).

6.2.1.2 Features to be tested

This test design will test the following features:

Features
Import start-up
Importing area folders structure
Ingestion Request File validity
Importing workflow
Ordering of Ingestion Request File
Test Ingestion Request File format
Import data files
Import header files
Import metadata files
Ingest queue

6.2.1.3 Approach refinements

The test cases used to verify this test design are:

- **IMPORT-BASIC-01** Start the Import server from the command line and verify that it runs successfully with no warnings or exception.



- IMPORT-BASIC-02 Verify that import area directories are successfully found.
- IMPORT-BASIC-03 Verify that Ingestion Request Files are successfully found and match the required Ingestion Request file pattern.
- IMPORT-BASIC-04 Verify that the processing of several Ingestion Request File is made based on the date that appears within the file name.
- IMPORT-BASIC-05 Verify that Ingestion Request Files in text format are handled correctly.
- IMPORT-BASIC-06 Verify that data files are transferred and identified (imported) correctly.
- IMPORT-BASIC-07 Verify that header files are transferred and identified (imported) correctly.
- IMPORT-BASIC-08 Verify that metadata files are transferred and identified (imported) correctly.
- IMPORT-BASIC-09 Verify that the import log associated to a file to be ingested has been associated correctly.

6.2.2 Test design *IMPORT-LOGGING*

6.2.2.1 General

This test design establishes procedures for the validation on the logging (reporting) capabilities of the Import software.

6.2.2.2 Features to be tested

This test design will test the following features:

Features
Log output
Report output

6.2.2.3 Approach refinements

The test cases used to verify this test design are:

- IMPORT-LOGGING-01 Test whether the logs are correctly written to the log file in the correct location.
- IMPORT-LOGGING-02 Verify that a report file has been produced (in future versions, the report file is sent to CAA and an email is sent too).

6.2.3 Test design *INGEST-BASIC*

6.2.3.1 General

This test design establishes procedures for the validation on the basic functionality of the ingest software. Being a largely automatic process, the success or otherwise of these tests can be determined from the output logs (verified in section 6.2.4).

6.2.3.2 Features to be tested

This test design will test the following features:

Features
Ingest start-up
Daemon process
Associated import logs
Ingest data files
Ingest XML metadata files
Ingest header files
Ingest inventory files
Ingest graphical metadata CSV files

6.2.3.3 Approach refinements

The test cases used to verify this test design are:

- INGEST-BASIC-01 Start the Ingest server from the command line and verify that it runs successfully with no warnings or exception.
- INGEST-BASIC-02 Verify that the daemon process will continue to run until explicitly killed.
- INGEST-BASIC-03 Verify that the import log associated to a file to be ingested has been associated correctly.
- INGEST-BASIC-04 Verify that data files can be ingested successfully.
- INGEST-BASIC-05 Verify that XML metadata files can be ingested successfully.
- INGEST-BASIC-06 Verify that header files can be ingested successfully.
- INGEST-BASIC-07 Verify that inventory files can be ingested successfully.
- INGEST-BASIC-08 Verify that graphical metadata files (CSV) can be ingested successfully.

6.2.4 Test design *INGEST-LOGGING*

6.2.4.1 General

This test design establishes procedures for the validation on the logging (reporting) capabilities of the Ingest software.



6.2.4.2 Features to be tested

This test design will test the following features:

Features
Log output
Report output for IRF sent by CAA
Report output for IRF generated at CSA

6.2.4.3 Approach refinements

The test cases used to verify this test design are:

- INGEST-LOGGING-01 Test whether the logs are correctly written to the log file in the correct location.
- INGEST-LOGGING-02 Verify that a report file has been produced and sent to CAA. Verify that an email has been sent too.
- INGEST-LOGGING-03 Verify that a report file has been produced.

6.2.5 Test design DD-BASIC

6.2.5.1 General

This test design establishes procedures for the validation on the basic functionality of the Data Distribution service.

6.2.5.2 Features to be tested

This test design will test the following features:

Features
Data Distribution service start-up
Port open

6.2.5.3 Approach refinements

The test cases used to verify this test design are:

- DD-BASIC-01 Start the Data Distribution service from the command line and verify that it has started successfully with no exceptions thrown.
- DD-BASIC-02 Verify that the Data Distribution service is running in the correct port for the particular deployment environment.

6.2.6 Test design DD-RETRIEVAL

6.2.6.1 General

This test design establishes procedures for the validation on the data retrieval capabilities of the Data Distribution service.

More detailed tests of the retrieval functionality can be found at the User Interface section, described in 6.2.13 Test design UI-RETRIEVAL, and to a lesser extent through the AIO, described in 6.2.23 Test design CAIO-RETRIEVAL.

6.2.6.2 Features to be tested

This test design will test the following features:

Features
Direct Download
Asynchronous download

6.2.6.3 Approach refinements

The test cases used to verify this test design are:

- DD-RETRIEVAL-01 Test that, for a direct download request, a URL is created from where the requested data can be downloaded successfully.
- DD-RETRIEVAL-02 Test that, for an asynchronous download request, an email is sent to user's email address containing a URL from where the requested data can be downloaded successfully.

6.2.7 Test design DD-LOGGING

6.2.7.1 General

This test design establishes procedures for the validation on the logging capabilities of the Data Distribution service.

6.2.7.2 Features to be tested

This test design will test the following features:

Features
Log output
Logging level

6.2.7.3 Approach refinements

The test cases used to verify this test design are:



- DD-LOGGING-01 Test whether the logs are correctly written to the log file in the correct location.
- DD-LOGGING-02 Test whether changing the logging level correctly changes the detail of logs provided.

6.2.8 Test design GRAPHICS-BASIC

6.2.8.1 General

This test design establishes procedures for the validation on the basic functionality of the Graphics module. This module makes use of Perl scripts to access CSA database to locate the source files for plotting, to call concatenation tool in Java to generate the source file and finally call IDL routines to generate the graphical plots.

6.2.8.2 Features to be tested

This test design will test the following features:

Features
Generation of on-demand plots
Generation of pre-generated plots

6.2.8.3 Approach refinements

The test cases used to verify this test design are:

- GRAPHICS-BASIC-01 Launch the Perl script that generates an on-demand plot from the command line and verify that it has run successfully with no exceptions thrown.
- GRAPHICS-BASIC-02 Launch the Perl script that generates a pre-generated plot from the command line and verify that it has run successfully with no exceptions thrown.

6.2.9 Test design GRAPHICS-LOGGING

6.2.9.1 General

This test design establishes procedures for the validation on the logging capabilities of the Graphics module.

6.2.9.2 Features to be tested

This test design will test the following features:

Features
Log output

Logging level

6.2.9.3 Approach refinements

The test cases used to verify this test design are:

- GRAPHICS-LOGGING-01 Test whether the logs are correctly written to the log file in the correct location.
- GRAPHICS-LOGGING-02 Test whether changing the logging level correctly changes the detail of logs provided.

6.2.10 Test design UI-BASIC

6.2.10.1 General

This test design establishes procedures for the validation on the very basic functionality aspects of the application like start-up, cross-platform support and the like.

6.2.10.2 Features to be tested

This test design will test the following features:

Features
Application start-up using Linux OS
Application start-up using Windows OS
Application start-up using Mac OS
Application start-up from outside ESAC firewall
General layout visible, readable and resizable
Menu and tool bar
Application exit

6.2.10.3 Approach refinements

The test cases used to verify this test design are:

- UI-BASIC-01 Start UI application in a Linux OS environment. Confirm that initial status bar advances while it displays text messages. Once the application has been loaded, check labels readability. Confirm that application window can be resized.
- UI-BASIC-02 Start UI application in a Windows OS environment. Confirm that initial status bar advances while it displays text messages. Once the application has been loaded, check labels readability. Confirm that application window can be resized.
- UI-BASIC-03 Start UI application in a Mac OS environment. Confirm that initial status bar advances while it displays text messages. Once the application has



been loaded, check labels readability. Confirm that application window can be resized.

- UI-BASIC-04 Start UI application outside of ESAC firewall (with any OS environment). Confirm that initial status bar advances while it displays text messages. Once the application has been loaded, check labels readability. Confirm that application window can be resized.
- UI-BASIC-05 Test whether basic menu and tool actions are working.
- UI-BASIC-06 Test application quits successfully and releases all resources effectively.

6.2.11 Test design UI-SEARCH

6.2.11.1 General

This test design establishes procedures for the validation on the search capabilities from the archive's user interface.

6.2.11.2 Features to be tested

This test design will test the following features:

Features
No data returned notification
Search for all datasets
Cancel search
Search for all inventory data
Search datasets with a given metadata criteria
Clear search
Search inventory data with a given criteria
Search fields input operators and validation

6.2.11.3 Approach refinements

The test cases used to verify this test design are:

- UI-SEARCH-01 Search for criteria known not to return any results and verify that a “No data returned” message is displayed.
- UI-SEARCH-02 Search for all datasets with no constraint and check number of matching results is correct.
- UI-SEARCH-03 Search for datasets, check that progress bar and mouse indicates progress, cancel and that the Cancel button stops a search successfully.
- UI-SEARCH-04 Search for all inventory records with no constraint and check number of matching results is correct.
- UI-SEARCH-05 Search for datasets matching a global criterion and check number of matching results is correct.



- UI-SEARCH-06 Clear search panels and fields, search for all datasets and check number of matching results is correct.
- UI-SEARCH-07 Search for inventory records matching a global criterion and check number of matching results is correct.
- UI-SEARCH-08 Search for datasets matching a criterion where logical operators and wildcards are used and check number of matching results is correct. Test also validation of input field values.

6.2.12 Test design UI-RESULTS

6.2.12.1 General

This test design establishes procedures for the validation on the metadata results display capabilities from the archive's user interface. Such features allow the user to select which datasets to display, to introduce a different time constraint or to select the way to display the datasets.

6.2.12.2 Features to be tested

This test design will test the following features:

Features
Short-list of datasets results display
Results details display
Results pagination
Results dynamic page size selection
Change time constraint for results
Change dataset units selection
Clear datasets selected
Save datasets in a profile
Select format of the files to deliver
Add/remove results to shopping basket
Shopping Basket overview results display
Shopping Basket results display
Docking/undocking of details windows
Docking/undocking of results tabs

6.2.12.3 Approach refinements

The test cases used to verify this test design are:

- UI-RESULTS-01 Search for datasets which belongs to the short-list criteria, then test whether the matching results are the correct ones. Unselect the short-list check box and check that the datasets displayed are the same as the ones obtained with the search for all datasets.



- UI-RESULTS-02 Search for all datasets. Select the magnifying glass for experiment and test that a separated details panel opens. Select a datasets name and test that a separated details panel opens too.
- UI-RESULTS-03 From the search results, test results pagination is working.
- UI-RESULTS-04 From the search results, change page size and test results are displayed accordingly.
- UI-RESULTS-05 Update time range for search and check number of matching results is correct.
- UI-RESULTS-06 On a category that contains drop-down list with different units, change the selected unit and check that the corresponding dataset Ids have changed accordingly.
- UI-RESULTS-07² From the search results, select some datasets and check that after use of the clear button, all selections have disappeared.
- UI-RESULTS-08² From the search results, select some datasets and save them into a General profile. Finally verify its functioning.
- UI-RESULTS-09² From the search results, select some datasets and check that the format drop-down list is working.
- UI-RESULTS-10² From the search results, select some datasets and test whether they can be sent to the shopping basket effectively. From the shopping basket overview check whether items added are displayed accordingly. From the shopping basket check whether items added can be deleted and test display globally.
- UI-RESULTS-11 Test docking and undocking of details windows.
- UI-RESULTS-12 Test docking and undocking of results tab.

6.2.13 Test design UI-RETRIEVAL

6.2.13.1 General

This test design establishes procedures for the validation on the data retrieval capabilities from the archive's user interface.

6.2.13.2 Features to be tested

This test design will test the following features:

Features
Direct download of cluster files request
Direct download of non-cluster files request
Shopping Basket of cluster files request
Shopping Basket of non-cluster files request

² This feature requires login and will also be tested from the security perspective later

6.2.13.3 Approach refinements

The test cases used to verify this test design are:

- UI-RETRIEVAL-01³ From the search results, select a dataset with cluster files, select desired format delivery (CEF or CDF), select desired delivery interval and test whether the data request can be downloaded locally to disk.
- UI-RETRIEVAL-02³ From the search results, select a dataset with non-cluster files, and test whether the data request can be downloaded locally to disk.
- UI-RETRIEVAL-03³ From the shopping basket, select a dataset with cluster files, select desired format delivery (CEF or CDF), select desired delivery interval and checkout the data request.
- UI-RETRIEVAL-04³ From the shopping basket, select a dataset with non-cluster files, and checkout the data request.

6.2.14 Test design UI-PLOTTING

6.2.14.1 General

This test design establishes procedures for the validation on the plotting capabilities of graphical products from the archive's user interface.

6.2.14.2 Features to be tested

This test design will test the following features:

Features
Selection of time interval
Selection of plot type
Selection of graphical products and sorting
Save Graphical products in a Graphical profile
Plot with no data
Display of plot in a separated window
Stack of plots and navigation
Plot next/previous product in time
Docking/undocking of plots windows
Retrieve plot in Postscript format
Retrieve of CEF source files for on-demand
Retrieve plot in PNG format for pre-generated

6.2.14.3 Approach refinements

The test cases used to verify this test design are:

³ This feature requires login and will also be tested from the security perspective later



- UI-PLOTTING-01 Enter time interval. Test validation of Date Range field.
- UI-PLOTTING-02 Select different Plot Types. Check that time interval changes for pre-generated plots and end time and duration cannot be modified by the user.
- UI-PLOTTING-03 Select different Graphical products to plot. Check that it is possible to decide the order of the products within the plot.
- UI-PLOTTING-04 Select some Graphical products and save them into a Graphical profile. Finally verify its functioning
- UI-PLOTTING-05⁴ Search for graphical product known not to return any results for a certain time and verify that an empty plot is displayed.
- UI-PLOTTING-06⁴ Plot one graphical product and test that a separated plot panel opens displaying the image. In addition, test that for an on-demand plot the maximum duration allowed is 57 hours.
- UI-PLOTTING-07⁴ Select more than 7 graphical products and generate and on-demand plot. Test that the resulting plots are stacked in the order specified and that a maximum of 7 plots are displayed per page. Test that it is possible to navigate to the next and previous page or previous graphical products plotted.
- UI-PLOTTING-08⁴ Perform a plot of a graphical product for a certain time range. Test that it is possible to perform the same plot with the same duration for the previous time range and for the next time range.
- UI-PLOTTING-09 Test docking and undocking of plots windows. In addition, test that the plot window can be detached in a stand-alone window and that the plots window can be opened from Windows menu.
- UI-PLOTTING-10⁴ From the plots window, test that the current displayed image can be downloaded in Postscript format.
- UI-PLOTTING-11⁴ From the plots window, test that the CEF source files of the current displayed on-demand plot can be downloaded.
- UI-PLOTTING-12⁴ From the plots window, test that the current displayed pre-generated image can be downloaded in PNG format.

6.2.15 Test design UI-SECURITY

6.2.15.1 General

This test design establishes procedures for the validation on the functionalities related to authentication and authorisation processes in place to access the data securely from the archive's user interface.

6.2.15.2 Features to be tested

This test design will test the following features:

⁴ This feature requires login and will also be tested from the security perspective later

Features

Login/logout
 Register a new user
 Access to public services
 Access to non-public services

6.2.15.3 Approach refinements

The test cases used to verify this test design are:

- UI-SECURITY-01 Test user authentication on the RSSD LDAP system. Check login/logout functionalities are working.
- UI-SECURITY-02 Test whether selecting to register a new user opens the registration page on the RSSD LDAP system in a browser window.
- UI-SECURITY-03 Test whether a user who has not logged in has only access to search for inventory data, search for datasets, and search for graphical products.
- UI-SECURITY-04 Test the user is requested to login when attempting to select a dataset, to plot a graphical product or want to access its profiles.

6.2.16 Test design UI-LOGGING

6.2.16.1 General

This test design establishes procedures for the validation on the logging functionalities of the archive's user interface.

6.2.16.2 Features to be tested

This test design will test the following features:

Features

Log output
 Logging levels

6.2.16.3 Approach refinements

The test cases used to verify this test design are:

- UI-LOGGING-01 Test whether the Log Console correctly displays the logging output as actions are performed in the user interface.
- UI-LOGGING-02 Test whether changing the logging level correctly changes the details of logs provided.

6.2.17 Test design UI-PERFORMANCE

6.2.17.1 General

The performance tests are designed to determine the performance of the system, comparing it to the expected one. Requirements specified in [CFA_EST_URD] shall be used when designing the performance tests cases and also when validating the test successfulness. These tests will basically focus on analysing the response time obtained in several scenarios designated previously as functionality tests, to assess if they meet the performance requirements specified in the Requirements Baseline [CFA_EST_URD]. Furthermore, some other tests will measure the capability of the system for managing a determined magnitude of load.

The scope of this test design is to prove that the system complies with the performance requirements specified in [CFA_EST_URD], and although these tests are always referred to the server layer onwards (data distribution, query manager and database), the time taken by the user interface to display results must be considered, though it may be disregarded sometimes due to pagination features, that reduce the amount of entries that are painted on the user interface.

All performance tests must run with caching capabilities disabled. This is because current JUnit tests have the queries hard-coded (get all datasets for a concrete experiment, get all inventory data for a certain experiment, and always the same experiment) and they are not randomly chosen. This might be changed in the future. However, for certain tests, a comparison between the same test case both with and without cache might be performed, to find out the real benefits of results caching.

For each performance test, the average time for 100 runs will be taken and the standard deviation will also be written down. Furthermore, each performance test will be checked against the performance requirements that it may be affected by (shown in the table below). The test will be marked as PASSED, only if it accomplishes its performance requirements (and of course without failing itself).

For some performance tests specified in the test case itself, a very light client and a powerful client will be used to find out if those functionalities are affected by the type of client and to what extent.

6.2.17.2 Features to be tested

This test design will test the following features:

Features
Web interface initialisation
Get all datasets list
Get all inventory data
Direct Download of data files
Retrieval of CEF products from Shopping Basket

Retrieval of non-CEF products from Shopping-Basket
 Plot of graphical products
 Log-in
 Availability of requested files

6.2.17.3 Approach refinements

The test cases used to verify this test design are:

- UI-PERFORMANCE-01 Test initialisation of the web interface.
- UI-PERFORMANCE-02 Get all datasets list.
- UI-PERFORMANCE-03 Get all inventory data.
- UI-PERFORMANCE-04 Direct Download of data files.
- UI-PERFORMANCE-05 Retrieve one CEF data file from the Shopping Basket (asynchronously).
- UI-PERFORMANCE-06 Retrieve one non-CEF data file from the Shopping Basket (asynchronously).
- UI-PERFORMANCE-07 Perform the plotting a graphical product.
- UI-PERFORMANCE-08 Perform a log-in to the system.
- UI-PERFORMANCE-09 Check that any retrieval request is available for download during one week.

6.2.18 Test design UI-SCALABILITY

6.2.18.1 General

Scalability tests aim at verifying that the system can scaled up to the engineering limits defined in the Requirements Baseline ([CFA_EST_URD]). The tests of this section will concentrate on testing the limit of the system, which can be defined as the magnitude of demand that can be placed on the system while continuing to meet latency and throughput requirements.

The idea is to get much our system scales up (maximum complex queries, maximum concurrent users, etc...), but always complying with the performance requirements defined.

6.2.18.2 Features to be tested

This test design will test the following features:

Features

Scalability analyses of queries
 Scalability analyses of retrievals
 Scalability analyses of typical workload scenarios

6.2.18.3 Approach refinements

The test cases used to verify this test design are:

- UI-SCALABILITY-01 Scalability analysis of a query that gets all datasets.
- UI-SCALABILITY-02 Scalability analysis of a query that gets all inventory data.
- UI-SCALABILITY-03 Scalability analysis of a query that gets datasets constrained in time.
- UI-SCALABILITY-04 Scalability analysis of a query that gets all datasets with filters applied.
- UI-SCALABILITY-05 Scalability analysis of a query that gets all inventory data with filters applied.
- UI-SCALABILITY-06 Scalability analysis of the retrieval of data close to the maximum size allowed for direct download (1Gb).
- UI-SCALABILITY-07 Scalability analysis of the retrieval of data close to the maximum size allowed for Shopping Basket (40Gb).
- UI-SCALABILITY-08 Scalability analysis of the plotting a graphical product for the four spacecraft available.
- UI-SCALABILITY-09 Scalability analysis for a typical workload scenario (i.e. 70% of queries, 29% of retrievals) for queries and retrievals with short time range.
- UI-SCALABILITY-10 Scalability analysis for a typical workload scenario (i.e. 70% of queries, 29% of retrievals) for queries and retrievals with large time range.

6.2.19 Test design UI-STRESS

6.2.19.1 General

The goal of stress testing is to evaluate and determine the behaviour of the software system while the load is over its designed capacity. The best way to design the stress tests is to impose the maximum limits on all system performance characteristics at the same time (response time, availability and throughput), which will provide with a set of worst-case conditions under which the system is still expected to operate acceptably well (although performance requirements may not be met). These tests ensure that the system performs acceptably under an expected peak load. Furthermore, system recovery must be assessed during these tests to prove that the system comes back to normal operation after a system failure (sometimes triggered by a huge load over the maximum limits) and that the cause of the failure can be identified.

This section gathers not only stress testing (under a short peak of load) but also stability testing. For accomplishing this goal, a test will be considered successful when it has run for a reasonable period of time (sometime more than a couple of hours).

These tests will be performed after the scalability tests, in order to assess the amount of load under the system still performs acceptably well, so that it can be increased to find out metrics for availability, stability and recovery capabilities.

6.2.19.2 Features to be tested

This test design will test the following features:

Features
Stressing the server layer with typical workload scenarios
Study recoverability of the system after a crash

6.2.19.3 Approach refinements

The test cases used to verify this test design are:

- UI-STRESS-01 Stress the server layer onwards with a typical workload scenario (70% of queries, 29% of retrievals) for different queries and retrievals with a short/larger time ranges given the results of the scalability tests.
- UI-STRESS-02 Study the recoverability of the system when it has crashed due to a huge amount of load as well as its response to possible DoS attacks.

6.2.20 Test design CAIO-BASIC

6.2.20.1 General

This test design establishes procedures for the validation on the very basic functionality aspects of the CAIO such as page loading and display, cross-platform support and the like.

6.2.20.2 Features to be tested

This test design will test the following features:

Features
Web pages load and display correctly in common browsers
Web pages load and display correctly using Linux OS
Web pages load and display correctly using Windows OS
Web pages load and display correctly using Mac OS
Web pages load and display correctly from outside ESAC firewall
Menu bar functionality
Links are correct
Popups notifications dealt with properly
AIO command line client runs correctly on Linux OS
AIO command line client runs correctly on Windows OS
AIO command line client runs correctly on Mac OS
Authentication via 'wget'

6.2.20.3 Approach refinements

The test cases used to verify this test design are:

- CAIO-BASIC-01 In a Linux OS environment, open the CAIO main web pages in each of the most common web browsers available. Verify that the pages open and display correctly and the CSS is adhered to.
- CAIO-BASIC-02 In a Windows OS environment, open the CAIO main web pages in each of the most common web browsers available. Verify that the pages open and display correctly and the CSS is adhered to.
- CAIO-BASIC-03 In a Mac OS environment, open the CAIO main web pages in each of the most common web browsers available. Verify that the pages open and display correctly and the CSS is adhered to.
- CAIO-BASIC-04 Open the main CAIO page (any browser, any environment) from outside the ESAC firewall. Verify that all pages open and display correctly and the CSS is adhered to.
- CAIO-BASIC-05 Navigate through the CAIO pages using the main menu bar and verify the correct web pages are opened.
- CAIO-BASIC-06 Test that every link provided on the CAIO web pages point to somewhere sensible.
- CAIO-BASIC-07 Make a HTTP request (product request) to the CAIO from each of the most common browsers. Test that any notifications concerning popup blocking are displayed correctly and that the popup itself displays correctly.
- CAIO-BASIC-08 In a Linux OS environment, make a query using the AIO command line client and verify that the action is completed successfully.
- CAIO-BASIC-09 In a Windows OS environment, make a query using the AIO command line client and verify that the action is completed successfully.
- CAIO-BASIC-10 In a Mac OS environment, make a query using the AIO command line client and verify that the action is completed successfully.
- CAIO-BASIC-11 Make a login request using 'wget' and store the credentials in a cookie, that can be used for further request via 'wget'.

6.2.21 Test design CAIO-SEARCH

6.2.21.1 General

This test design establishes procedures for the validation on the ability to build queries and perform the relevant searches through the CAIO subsystem.

A thorough test of every expected type of query combination should be performed through an automatic process in which the outcome of a pre-prepared set of queries is validated.

6.2.21.2 Features to be tested

This test design will test the following features:

Features
No data returned notification from product queries
No metadata returned notification from metadata queries
Build all possible metadata queries

Build all possible product request (synchronous and asynchronous)
 Use of RETURN_TYPE parameter for metadata
 Specifying PAGE_SIZE and PAGE for metadata
 Using the complex QUERY parameter for metadata
 Specifying ORDER_BY for metadata
 Specifying RETRIEVALTYPE for certain product queries
 Badly-formed query notifications

6.2.21.3 Approach refinements

The test cases used to verify this test design are:

- CAIO-SEARCH-01 Create a product data query for a product known not to exist and verify a relevant notification is displayed explaining why the data cannot be retrieved.
- CAIO-SEARCH-02 Create a metadata query known not to return any results and verify that a relevant notification is displayed explaining that no metadata was found.
- CAIO-SEARCH-03 Query every type of metadata combination and verify the correct number and type of results is returned.
- CAIO-SEARCH-04 Query every type of permissible product data and verify the correct file is returned.
- CAIO-SEARCH-05 For any metadata query specify a RETURN_TYPE of VOTABLE, JSON, CSV and verify the returned result is in the correct format.
- CAIO-SEARCH-06 For a metadata query with many results, specify PAGE_SIZE and PAGE and verify that the correct number of results is returned.
- CAIO-SEARCH-07 Build a metadata query using the QUERY= parameter and verify the correct results are returned.
- CAIO-SEARCH-08 For a metadata query, specify an order using ORDER_BY and verify the results are appropriately ordered.
- CAIO-SEARCH-09 For a header data query, specify the RETRIEVALTYPE= HEADER parameter and verify the returned product is a XML header description.
- CAIO-SEARCH-10 Verify that creating a bad query will prompt a relevant notification explaining the fact to the user.

6.2.22 Test design CAIO-RESULTS

6.2.22.1 General

This test design establishes procedures for the validation on the metadata results display in the various file formats for user defined queries as well as the display of the results pre-defined Cluster datasets metadata queries.

6.2.22.2 Features to be tested

This test design will test the following features:

Features

Results display as VOTable
 Results display in JSON
 Results display in CSV
 Pagination
 Cluster datasets results display
 Retrieval links

6.2.22.3 Approach refinements

The test cases used to verify this test design are:

- CAIO-RESULTS-01 For a metadata query with RETURN_TYPE of VOTable, verify that the returned results are in VOTable format and the correct number of results and pagination are returned. Verify that any relevant retrieval links exist and work correctly.
- CAIO-RESULTS-02 For a metadata query with RETURN_TYPE of JSON, verify that the returned results are in JSON format and the correct number of results and pagination are returned. Verify that any relevant retrieval links exist and work correctly.
- CAIO-RESULTS-03 For a metadata query with RETURN_TYPE of CSV, verify that the returned results are in CSV format and the correct number of results and pagination are returned.

6.2.23 Test design CAIO-RETRIEVAL

6.2.23.1 General

This test design establishes procedures for the validation on the data retrieval capabilities of the CAIO.

6.2.23.2 Features to be tested

This test design will test the following features:

Features

Retrieve result of data product request synchronously
 Retrieve header of data request
 Retrieve result from several datasets synchronously
 Retrieve inventory data
 Retrieve result of data product request asynchronously
 Retrieve result from several datasets synchronously
 Retrieve the user manual from web pages
 Retrieve the AIO client from web pages
 Retrieve metadata file in a given format

6.2.23.3 Approach refinements

The test cases used to verify this test design are:

- CAIO-RETRIEVAL-01 Verify that the result of a synchronous data product request starts the download of the correct file successfully.
- CAIO-RETRIEVAL-02 Verify that the result of a data request, with REQUESTTYPE=HEADER specified, starts the download of the correct header XML file successfully.
- CAIO-RETRIEVAL-03 Verify that the result of a synchronous product data request for a group of datasets starts the download of a single compressed file containing the grouped files successfully.
- CAIO-RETRIEVAL-04 Verify that specifying DATASET_INVENTORY for a metadata request starts the download of the correctly associated inventory data.
- CAIO-RETRIEVAL-05 Verify that the result of an asynchronous data product request starts the download of the correct file successfully.
- CAIO-RETRIEVAL-06 Verify that the result of an asynchronous product data request for a group of datasets starts the download of a single compressed file containing the grouped files successfully.
- CAIO-RETRIEVAL-07 Verify that the User Manual can be downloaded from the CAIO web pages successfully and that it is the most recent version.
- CAIO-RETRIEVAL-08 Verify that the AIO command line client example can be downloaded from the CAIO web pages successfully and that it is the most recent version.
- CAIO-RETRIEVAL-09 Verify that metadata can be successfully downloaded for each of the specified file formats: VOTable, JSON, CSV.

6.2.24 Test design CAIO-LOGGING

6.2.24.1 General

This test design establishes procedures for the validation on the logging functionalities of the CAIO.

6.2.24.2 Features to be tested

This test design will test the following features:

Features
Log output
Logging levels

6.2.24.3 Approach refinements

The test cases used to verify this test design are:



- CAIO-LOGGING-01 Test whether the logs are correctly written to the log file in the location of the webapps files for the relevant deployment environment.
- CAIO-LOGGING-02 Test whether changing the logging level correctly changes the details of logs provided.

6.2.25 Test design CAIO-SECURITY

6.2.25.1 General

This test design establishes procedures for the validation on the functionalities related to authentication and authorisation processes in place to access the data securely from the CAIO.

6.2.25.2 Features to be tested

This test design will test the following features:

Features
Login/logout
Access to public services
Access to non-public services

6.2.25.3 Approach refinements

The test cases used to verify this test design are:

- CAIO-SECURITY-01 Test user authentication on the RSSD LDAP system. Check login/logout functionalities are working.
- CAIO-SECURITY-02 Test whether a user who has not logged in has only access to perform metadata requests (including inventory data requests).
- CAIO-SECURITY-03 Test the user is requested to login when attempting to perform a synchronous or asynchronous data request.

7 SOFTWARE VALIDATION TEST SPECIFICATION

7.1 General

The specification of the testing cases follows the testing specification design described in section 6. The naming convention for the test cases is the test design (CI acronym followed by the area identifier) and two digits denoting the test case number for the relevant test design.



For those test cases in which the Input Specifications consist on different actions, these have been numbered, thus the matching output has the same number within the Output Specifications.

In the preparation of the test cases, special attention has been paid to regression tests coverage. A few recommended guidelines to facilitate regression testing are the following:

- Test procedures should be prepared to run automatically whenever possible.
- Test procedures should produce output files to ease automatic comparison during regression tests.
- A fix set of input data and its corresponding known output should be kept among releases in order to be reused for further comparison purposes.

As mention before, a traceability matrix User Requirements – Test Cases can be found at [CSA_URD_Test_Matrix]. Please note that, for this first version of the Software Validation Specification document, only those User Requirements with “Priority = 1” and “Need = 1” have been considered for the creation of Test Cases, with the following exceptions:

- a) URs priority 1, need 1 not considered because the functionality will not appear at the CSA Public Release 1.0:
 - Inventory plots user requirements: UR 3.6.1, UR 3.6.2, UR 3.6.3
- b) URs not priority 1 or not need 1 considered because the functionality is already included at the CSA Public Release 1.0:
 - User guide of the CSA system: UR 2.5.2
 - Context sensitive help at GUI: UR 3.1.2
 - Key datasets on top of the list: UR 3.1.15
 - Estimation of size to download: UR 3.2.4

7.2 IMPORT-BASIC Test Cases

7.2.1 Test Case *IMPORT-BASIC-01: Start-up of Import server*

This test validates that the Import server starts up correctly, provided no other instance is already running, and verifies that no warnings or exceptions are thrown.

7.2.1.1 Inputs specification

Start the Import server from command line.

7.2.1.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.2.1.3 Test pass-fail criteria

Test passed if the Import server starts successfully without throwing any warnings or exceptions, failed otherwise.



7.2.1.4 Environmental needs

None.

7.2.1.5 Special procedural constraints

Not applicable.

7.2.1.6 Interface dependencies

None.

7.2.2 Test Case *IMPORT-BASIC-02: Import directories available*

This test validates that the required directories exist and are found successfully.

7.2.2.1 Inputs specification

Not applicable.

7.2.2.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs (when the directories are not available error logs are written and Import server stops).

7.2.2.3 Test pass-fail criteria

Test passed if all the required import directories are found, failed otherwise.

7.2.2.4 Environmental needs

None.

7.2.2.5 Special procedural constraints

Not applicable.

7.2.2.6 Interface dependencies

Test case IMPORT-BASIC-01 starts the process.

7.2.3 Test Case *IMPORT-BASIC-03: Ingestion Request File validity*

This test validates that Ingestion Request Files are found successfully. It is also tested that the check against matches of the required file name format pattern is performed correctly.

7.2.3.1 Inputs specification

Not applicable.



7.2.3.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.2.3.3 Test pass-fail criteria

Test passed if Ingestion Request Files are found and match the require file name pattern, failed otherwise.

7.2.3.4 Environmental needs

None.

7.2.3.5 Special procedural constraints

Not applicable.

7.2.3.6 Interface dependencies

Test case IMPORT-BASIC-01 starts the process.

7.2.4 Test Case *IMPORT-BASIC-04: Sorting of Ingestion Request Files*

This test validates that the Import server correctly deal with several Ingestion Request Files by sorting them using the date at the Ingestion Request File name, the oldest the first to be processed.

7.2.4.1 Inputs specification

Not applicable.

7.2.4.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.2.4.3 Test pass-fail criteria

Test passed if when there are several Ingestion Request Files, the first to be processed is the one with the oldest date that appears within the file name, failed otherwise.

7.2.4.4 Environmental needs

None.

7.2.4.5 Special procedural constraints

Not applicable.

7.2.4.6 Interface dependencies

Test case IMPORT-BASIC-01 starts the process.

7.2.5 Test Case *IMPORT-BASIC-05: Ingestion Request File format*

This test validates that an Ingestion Request File which is in text format can be parsed and imported correctly.

7.2.5.1 Inputs specification

Not applicable.

7.2.5.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.2.5.3 Test pass-fail criteria

Test passed if Ingestion Request Files in text format can be parsed and imported correctly, failed otherwise.

7.2.5.4 Environmental needs

None.

7.2.5.5 Special procedural constraints

Not applicable.

7.2.5.6 Interface dependencies

Test case IMPORT-BASIC-01 starts the process.

7.2.6 Test Case *IMPORT-BASIC-06: Import Data files*

This test validates that the Import server correctly identifies Data Files and their corresponding XML metadata files and subsequently imports the relevant Data and XML files successfully. Additionally, this test verifies that the import will fail if the files do not match the necessary schema agreed in the ICD.

7.2.6.1 Inputs specification

An Ingestion Request File containing Data files and the corresponding XML files is provided.

7.2.6.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.



7.2.6.3 Test pass-fail criteria

Test passed if Data files and their associated XML files are correctly identified and imported, failed otherwise.

7.2.6.4 Environmental needs

None.

7.2.6.5 Special procedural constraints

Not applicable.

7.2.6.6 Interface dependencies

Test case IMPORT-BASIC-01 starts the process.

7.2.7 Test Case IMPORT-BASIC-07: Import Header files

This test validates that the Import server correctly identifies Header files and subsequently imports the relevant Header files successfully. Additionally, this test verifies that the import will fail if the files do not match the necessary schema agreed in the ICD.

7.2.7.1 Inputs specification

An Ingestion Request File containing Header files is provided.

7.2.7.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.2.7.3 Test pass-fail criteria

Test passed if Header files are correctly identified and imported, failed otherwise.

7.2.7.4 Environmental needs

None.

7.2.7.5 Special procedural constraints

Not applicable.

7.2.7.6 Interface dependencies

Test case IMPORT-BASIC-01 starts the process.

7.2.8 Test Case IMPORT-BASIC-08: Import Metadata files

This test validates that the Import server correctly identifies Metadata files and subsequently imports the relevant Metadata files successfully. Additionally, this test



verifies that the import will fail if the files do not match the necessary schema agreed in the ICD.

7.2.8.1 Inputs specification

An Ingestion Request File containing Metadata files (XML files for mission, experiment, observatory, instrument, dataset, parameter, inventory, graphical product,...) is provided.

7.2.8.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.2.8.3 Test pass-fail criteria

Test passed if Metadata files are correctly identified and imported, failed otherwise.

7.2.8.4 Environmental needs

None.

7.2.8.5 Special procedural constraints

Not applicable.

7.2.8.6 Interface dependencies

Test case IMPORT-BASIC-01 starts the process.

7.2.9 Test Case IMPORT-BASIC-09: Ingest queue

This test validates that files pending ingestion are correctly identified and added to the ingest queue.

7.2.9.1 Inputs specification

Not applicable.

7.2.9.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs and database.

7.2.9.3 Test pass-fail criteria

Test passed if files pending ingestion are correctly added to the ingest queue, failed otherwise.

7.2.9.4 Environmental needs

None.



7.2.9.5 Special procedural constraints

Not applicable.

7.2.9.6 Interface dependencies

Test case IMPORT-BASIC-01 starts the process.

7.3 IMPORT-LOGGING Test Cases

7.3.1 Test case IMPORT-LOGGING-01: Log output

This test validates whether the logs are correctly written to the log file in the correct location.

7.3.1.1 Inputs specification

Go to the location for the import log files.

7.3.1.2 Outputs specification

A log file for the Import process exists.

7.3.1.3 Test pass-fail criteria

Test passed if the logs are correctly written in the correct location, failed otherwise.

7.3.1.4 Environmental needs

Not applicable.

7.3.1.5 Interface dependencies

None.

7.3.2 Test case IMPORT-LOGGING-02: Report Output

This test validates that for every Ingestion Request File processed a report is written indicating the exit status of the processing.

7.3.2.1 Inputs specification

Provide several Ingestion Request Files to the Import process.

7.3.2.2 Outputs specification

A report file per Ingestion Request File is written containing information about the status of the processing (failed or success) and the time it took to process it.



7.3.2.3 Test pass-fail criteria

Test passed if a report file per Ingestion Request file is written and sent to CAA, failed otherwise.

7.3.2.4 Environmental needs

Not applicable.

7.3.2.5 Interface dependencies

None.

7.4 INGEST-BASIC Test Cases

7.4.1 Test Case *INGEST-BASIC-01: Start-up of Ingest server*

This test validates that the Ingest server starts up correctly, provided no other instance is already running, and verifies that no warnings or exceptions are thrown.

7.4.1.1 Inputs specification

Start the Ingest server from command line.

7.4.1.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.4.1.3 Test pass-fail criteria

Test passed if the Ingest server starts successfully without throwing any warnings or exceptions, failed otherwise.

7.4.1.4 Environmental needs

None.

7.4.1.5 Special procedural constraints

Not applicable.

7.4.1.6 Interface dependencies

None.

7.4.2 Test Case *INGEST-BASIC-02: Daemon process*

This test validates that the Ingest daemon process will continue to run until the kill command is explicitly requested.



7.4.2.1 Inputs specification

Start the Ingest Daemon process from command line. Execute kill command.

7.4.2.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.4.2.3 Test pass-fail criteria

Test passed if the Ingest daemon runs correctly and stops when the kill command is requested, failed otherwise.

7.4.2.4 Environmental needs

None.

7.4.2.5 Special procedural constraints

Not applicable.

7.4.2.6 Interface dependencies

None.

7.4.3 Test Case *INGEST-BASIC-03: Associated import logs*

This test validates that a file pending ingestion has the correct Import log associated to it.

7.4.3.1 Inputs specification

Not applicable.

7.4.3.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs or database tables.

7.4.3.3 Test pass-fail criteria

Test passed if the file pending ingestion has the correct Import log (at database) associated to it, failed otherwise.

7.4.3.4 Environmental needs

None.

7.4.3.5 Special procedural constraints

Not applicable.



7.4.3.6 Interface dependencies

Test case INGEST-BASIC-01 or INGEST-BASIC-02 starts the process.

7.4.4 Test Case *INGEST-BASIC-04: Ingest Data files*

This test validates that Data files are correctly identified and all the relevant metadata correctly ingested.

7.4.4.1 Inputs specification

Not applicable.

7.4.4.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.4.4.3 Test pass-fail criteria

Test passed if Data files are correctly ingested, failed otherwise.

7.4.4.4 Environmental needs

None.

7.4.4.5 Special procedural constraints

Not applicable.

7.4.4.6 Interface dependencies

Test case IMPORT-BASIC-06 must be executed.

Test case INGEST-BASIC-01 or INGEST-BASIC-02 starts the process.

7.4.5 Test Case *INGEST-BASIC-05: Ingest XML metadata files*

This test validates that XML metadata files are correctly identified and all the relevant metadata correctly ingested. XML metadata files can contain metadata about mission, experiment, observatory, instrument, dataset and parameter entities.

7.4.5.1 Inputs specification

Not applicable.

7.4.5.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.



7.4.5.3 Test pass-fail criteria

Test passed if XML metadata files are correctly ingested, failed otherwise.

7.4.5.4 Environmental needs

None.

7.4.5.5 Special procedural constraints

Not applicable.

7.4.5.6 Interface dependencies

Test case IMPORT-BASIC-08 must be executed.

Test case INGEST-BASIC-01 or INGEST-BASIC-02 starts the process.

7.4.6 Test Case *INGEST-BASIC-06: Ingest Header files*

This test validates that Header files are correctly identified and all the relevant metadata correctly ingested. Header files can contain metadata about mission, experiment, observatory, instrument, dataset and parameter entities or be just caveats.

7.4.6.1 Inputs specification

Not applicable.

7.4.6.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.4.6.3 Test pass-fail criteria

Test passed if Header files are correctly ingested, failed otherwise.

7.4.6.4 Environmental needs

None.

7.4.6.5 Special procedural constraints

Not applicable.

7.4.6.6 Interface dependencies

Test case IMPORT-BASIC-07 must be executed.

Test case INGEST-BASIC-01 or INGEST-BASIC-02 starts the process.



7.4.7 Test Case *INGEST-BASIC-07: Ingest Inventory files*

This test validates that XML Inventory files are correctly identified and all the relevant metadata correctly ingested.

7.4.7.1 Inputs specification

Not applicable.

7.4.7.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.4.7.3 Test pass-fail criteria

Test passed if XML inventory files are correctly ingested, failed otherwise.

7.4.7.4 Environmental needs

None.

7.4.7.5 Special procedural constraints

Not applicable.

7.4.7.6 Interface dependencies

Test case IMPORT-BASIC-08 must be executed.

Test case INGEST-BASIC-01 or INGEST-BASIC-02 starts the process.

7.4.8 Test Case *INGEST-BASIC-08: Ingest Graphical Metadata CSV files*

This test validates that CSV graphical metadata files are correctly identified and all the relevant metadata correctly ingested. CSV graphical metadata files contain metadata about pregenerated plots files.

7.4.8.1 Inputs specification

Not applicable.

7.4.8.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.4.8.3 Test pass-fail criteria

Test passed if CSV graphical metadata files are correctly ingested and removed from disk, failed otherwise.

7.4.8.4 Environmental needs

None.

7.4.8.5 Special procedural constraints

Not applicable.

7.4.8.6 Interface dependencies

Test case IMPORT-BASIC-o8 must be executed.

Test case INGEST-BASIC-01 or INGEST-BASIC-02 starts the process.

7.5 INGEST-LOGGING Test Cases

7.5.1 *Test case INGEST-LOGGING-01: Log output*

This test validates whether the logs are correctly written to the log file in the correct location.

7.5.1.1 Inputs specification

Go to the location for the ingest log files.

7.5.1.2 Outputs specification

A log file for the Ingest process or daemon exists.

7.5.1.3 Test pass-fail criteria

Test passed if the logs are correctly written in the correct location, failed otherwise.

7.5.1.4 Environmental needs

Not applicable.

7.5.1.5 Interface dependencies

None.

7.5.2 *Test case INGEST-LOGGING-02: Report Output for Ingestion Request Files sent by CAA*

This test validates that for every Ingestion Request File processed a report is written.

The report contains the list of files processed indicating if the ingestion was done or it failed.

7.5.2.1 Inputs specification

Provide several Ingestion Request Files to the Ingest process.



7.5.2.2 Outputs specification

A report file per Ingestion Request File is written containing information about the status of the processing (failed or success) and the time it took to process it.

7.5.2.3 Test pass-fail criteria

Test passed if a report file per Ingestion Request file is written and sent to CAA, failed otherwise.

7.5.2.4 Environmental needs

Not applicable.

7.5.2.5 Interface dependencies

Test case IMPORT-LOGGING-02 may be used as input.

7.5.3 *Test case INGEST-LOGGING-03: Report Output for Ingestion Request Files generated at CSA*

This test validates that for every Ingestion Request File generated at CSA (in order to ingest CSV graphical metadata) a report file is written. The report contains the list of files processed indicating if the ingestion was done or it failed.

7.5.3.1 Inputs specification

Provide several Ingestion Request Files generated at CSA to the Ingest process.

7.5.3.2 Outputs specification

A report file per Ingestion Request File is written containing information about the status of the processing (failed or success) and the time it took to process it.

7.5.3.3 Test pass-fail criteria

Test passed if a report file per Ingestion Request file is written, failed otherwise.

7.5.3.4 Environmental needs

Not applicable.

7.5.3.5 Interface dependencies

Test case IMPORT-LOGGING-02 may be used as input.



7.6 DD-BASIC Test Cases

7.6.1 Test Case DD-BASIC-01: Start-up of Data Distribution service

This test validates that the Data Distribution service starts up correctly, provided no other instance is already running in the same port, and verifies that exceptions are thrown.

7.6.1.1 Inputs specification

Start the Data Distribution server from command line.

7.6.1.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the output logs.

7.6.1.3 Test pass-fail criteria

Test passed if the Data Distribution service starts successfully without throwing any exceptions, failed otherwise.

7.6.1.4 Environmental needs

None.

7.6.1.5 Special procedural constraints

Not applicable.

7.6.1.6 Interface dependencies

None.

7.6.2 Test Case DD-BASIC-02: Port open for Data Distribution service

This test validates that the Data Distribution service is running in the correct port for the associated deployment environment.

7.6.2.1 Inputs specification

Start the Data Distribution server from command line.

7.6.2.2 Outputs specification

To be able to verify that this test has completed successfully it may be necessary to consult the deployment environment status.

7.6.2.3 Test pass-fail criteria

Test passed if the Data Distribution service runs in the correct port, failed otherwise.



7.6.2.4 Environmental needs

None.

7.6.2.5 Special procedural constraints

Not applicable.

7.6.2.6 Interface dependencies

None.

7.7 DD-RETRIEVAL Test Cases

7.7.1 Test Case DD-RETRIEVAL-01: Direct download service

This test validates that an URL is created and returned that allows requested data to be directly downloaded successfully when a direct download request is specified.

7.7.1.1 Inputs specification

Direct download request.

7.7.1.2 Outputs specification

A URL to access the requested data is returned.

7.7.1.3 Test pass-fail criteria

Test passed if the data can be downloaded directly successfully, failed otherwise.

7.7.1.4 Environmental needs

None.

7.7.1.5 Special procedural constraints

Not applicable.

7.7.1.6 Interface dependencies

None.

7.7.2 Test Case DD-RETRIEVAL-02: Asynchronous download service

This test validates that an email is sent to the specified email address containing an URL that allows the requested data to be downloaded successfully when an asynchronous download request is specified.



7.7.2.1 Inputs specification

Asynchronous download request.

7.7.2.2 Outputs specification

A URL to access the requested data is returned and an e-mail sent (if e-mail address is specified).

7.7.2.3 Test pass-fail criteria

Test passed if the data can be downloaded asynchronously successfully via the URL provided in an email (or in case of CAIO request returned after an asynchronous request), failed otherwise.

7.7.2.4 Environmental needs

None.

7.7.2.5 Special procedural constraints

Not applicable.

7.7.2.6 Interface dependencies

None.

7.8 DD-LOGGING Test Cases

7.8.1 Test case DD-LOGGING-01: Log output

This test validates whether the logs are correctly written to the log file in the correct location.

7.8.1.1 Inputs specification

Go to the location of the log files.

7.8.1.2 Outputs specification

A log file for the DD service exists.

7.8.1.3 Test pass-fail criteria

Test passed if the logs are correctly written in the correct location, failed otherwise.

7.8.1.4 Environmental needs

Not applicable.

7.8.1.5 Interface dependencies

None.

7.8.2 Test case *DD-LOGGING-02: Logging levels*

This test validates whether changing the logging level correctly changes the detail of logs provided.

7.8.2.1 Inputs specification

Change the log level at the DD from INFO to DEBUG.

7.8.2.2 Outputs specification

More detailed logging is performed.

7.8.2.3 Test pass-fail criteria

Test passed if detail of logging output correctly changes when different levels are specified, failed otherwise.

7.8.2.4 Environmental needs

Not applicable.

7.8.2.5 Interface dependencies

None.

7.9 GRAPHICS-BASIC Test Cases

7.9.1 Test Case *GRAPHIC-BASIC-01: Generate On-demand plot*

This test validates that the Perl script that generates an “on-demand” plot runs correctly, and verifies that exceptions are thrown.

7.9.1.1 Inputs specification

Launch the Perl script with criteria to generate an “on-demand” plot from command line.

7.9.1.2 Outputs specification

Plot has been generated, given as results at least one gif file, one ps file and the CEF source files used. To be able to complete the verification that this test has completed successfully it may be necessary to consult the output logs.

7.9.1.3 Test pass-fail criteria

Test passed if the script runs successfully without throwing any exceptions, failed otherwise.



7.9.1.4 Environmental needs

None.

7.9.1.5 Special procedural constraints

Not applicable.

7.9.1.6 Interface dependencies

None.

7.9.2 Test Case GRAPHIC-BASIC-02: Generate pre-generated plot

This test validates that the Perl script that generates a “pre-generated” plot runs correctly, and verifies that exceptions are thrown.

7.9.2.1 Inputs specification

Launch the Perl script with criteria to generate a “pre-generated” plot from command line.

7.9.2.2 Outputs specification

Plot has been generated, given as results one png file, one ps file and the CSV metadata file to ingest. To be able to complete the verification that this test has completed successfully it may be necessary to consult the output logs.

7.9.2.3 Test pass-fail criteria

Test passed if the script runs successfully without throwing any exceptions, failed otherwise.

7.9.2.4 Environmental needs

None.

7.9.2.5 Special procedural constraints

Not applicable.

7.9.2.6 Interface dependencies

None.

7.10 GRAPHICS-LOGGING Test Cases

7.10.1 Test case GRAPHICS-LOGGING-01: Log output

This test validates whether the logs are correctly written to the log file in the correct location.



7.10.1.1 Inputs specification

Go to the location of the log files.

7.10.1.2 Outputs specification

A log file for the Graphics module exists.

7.10.1.3 Test pass-fail criteria

Test passed if the logs are correctly written in the correct location, failed otherwise.

7.10.1.4 Environmental needs

Not applicable.

7.10.1.5 Interface dependencies

None.

7.10.2 Test case *GRAPHICS-LOGGING-02: Logging levels*

This test validates whether changing the logging level correctly changes the detail of logs provided.

7.10.2.1 Inputs specification

Change the log level at the Graphics module from INFO to DEBUG.

7.10.2.2 Outputs specification

More detailed logging is performed.

7.10.2.3 Test pass-fail criteria

Test passed if detail of logging output correctly changes when different levels are specified, failed otherwise.

7.10.2.4 Environmental needs

Not applicable.

7.10.2.5 Interface dependencies

None.

7.11 UI-BASIC Test Cases

7.11.1 Test Case UI-BASIC-01: Application start-up using Linux OS

This test case validates that the CSA User Interface application starts in a Linux OS environment. It checks for compatibilities problems with the browser and/or java web start installation. Additionally, it confirms that the application has been loaded, the labels are displayed correctly and the application window can be resized.

7.11.1.1 Inputs specification

- 1- User starts the CSA User Interface Java Web Start application.
- 2- Once the application has been loaded, user resizes the window.

7.11.1.2 Outputs specification

- 1- First, a loading window appears with a status bar that advances while it displays text messages.
- 2- Once the application has been loaded, the main window of CSA UI is displayed, and all the labels are readable. The main window is resizable.

7.11.1.3 Test pass-fail criteria

Test passed if application shows up and reacts upon resizing, failed otherwise.

7.11.1.4 Environmental needs

This test case shall be executed in a Linux OS environment. Java Web Start must be installed and available on the client machine.

7.11.2 Test Case UI-BASIC-02: Application start-up using Windows OS

This test case validates that the CSA User Interface application starts in a Windows OS environment. It checks for compatibilities problems with the browser and/or java web start installation. Additionally, it confirms that the application has been loaded, the labels are displayed correctly and the application window can be resized.

7.11.2.1 Inputs specification

- 1- User starts the CSA User Interface Java Web Start application.
- 2- Once the application has been loaded, user resizes the window.

7.11.2.2 Outputs specification

- 1- First, a loading window appears with a status bar that advances while it displays text messages.
- 2- Once the application has been loaded, the main window of CSA UI is displayed, and all the labels are readable. The main window is resizable.



7.11.2.3 Test pass-fail criteria

Test passed if application shows up and reacts upon resizing, failed otherwise.

7.11.2.4 Environmental needs

This test case shall be executed in a Solaris OS environment. Java Web Start must be installed and available on the client machine. If firewall “COMODO” is installed, it shall allow “javaws” execution.

7.11.3 Test Case UI-BASIC-03: Application start-up using Mac OS

This test case validates that the CSA User Interface application starts in a Mac OS environment. It checks for compatibilities problems with the browser and/or java web start installation. Additionally, it confirms that the application has been loaded, the labels are displayed correctly and the application window can be resized.

7.11.3.1 Inputs specification

- 1- User starts the CSA User Interface Java Web Start application.
- 2- Once the application has been loaded, user resizes the window.

7.11.3.2 Outputs specification

- 1- First, a loading window appears with a status bar that advances while it displays text messages.
- 2- Once the application has been loaded, the main window of CSA UI is displayed, and all the labels are readable. The main window is resizable.

7.11.3.3 Test pass-fail criteria

Test passed if application shows up and reacts upon resizing, failed otherwise.

7.11.3.4 Environmental needs

This test case shall be executed in a Mac OS environment. Java Web Start must be installed and available on the client machine.

7.11.4 Test Case UI-BASIC-04: Start-up from outside ESAC firewall

This test case validates that the CSA User Interface application starts in an environment outside of the ESAC firewall and under network restrictive environments where only standard HTTP port 80 is opened. Specifically, this test case validates the client-server connection from an external machine through port 80.

7.11.4.1 Inputs specification

- 1- User starts the CSA User Interface Java Web Start application.
- 2- A simple query on datasets is done (no criteria specified).



7.11.4.2 Outputs specification

“Data Request” view is displayed with all datasets available.

7.11.4.3 Test pass-fail criteria

Test passed if application shows up and queries can be done.

7.11.4.4 Environmental needs

The server machine should have port 8080 closed. Client machine must be outside ESAC firewall and with outgoing port 80 opened only (note: use for this ‘sattest01.esac.esa.int’, user ‘satops’).

7.11.5 Test Case UI-BASIC-05: Menu and Toolbar

This test validates the basic actions triggered from both the menu and the tool bars. The actions include:

- Reset default view
- Configure interface colour
- One-click access to display Search panel, Results panel, Shopping Basket panel, Graphical Products Search panel, Plots panel, Profiles panel, Log console and Help topics
- Links to helpdesk, register and reset password
- Access to memory monitor
- Memory cache clean-up

7.11.5.1 Inputs specification

User clicks in all menu items and in all toolbar buttons.

7.11.5.2 Outputs specification

Every item click starts an action.

7.11.5.3 Test pass-fail criteria

Test passed if application reacts upon actions above, failed otherwise.

7.11.5.4 Environmental needs

Not applicable.

7.11.5.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.



7.11.6 Test Case UI-BASIC-06: Application exit

This test validates that the shut-down of the CSA User Interface releases all resources effectively.

7.11.6.1 Inputs specification

- 1- User close main window or choose “Exit” action.
- 2- User confirms exit.

7.11.6.2 Outputs specification

- 1- A confirmation window is displayed.
- 2- After confirmation, the main window disappears.

7.11.6.3 Test pass-fail criteria

Test passed if application disappears after exiting and no resources are left running, failed otherwise.

7.11.6.4 Environmental needs

Not applicable.

7.11.6.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.12 UI-SEARCH Test Cases

7.12.1 Test case UI-SEARCH-01: No data returned notification

This test validates that, when searching using criteria that there are no results for, the application will display a “No data returned” message.

7.12.1.1 Inputs specification

Perform a datasets search with criteria for which there are not results (for example, in the future).

7.12.1.2 Outputs specification

An information window is displayed with a “No results returned” message.

7.12.1.3 Test pass-fail criteria

Test passed if the output is displayed, failed otherwise.

7.12.1.4 Environmental needs

Not applicable.



7.12.1.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.12.2 Test case UI-SEARCH-02: Search for all datasets

This test validates that performing a search of all available datasets returns the correct number of matching results.

7.12.2.1 Inputs specification

No criteria entered before performing the search.

7.12.2.2 Outputs specification

“Data Request” tab is displayed listing the 12 available experiments. Within every experiment all datasets belonging to it are listed ordered by display order. The datasets within the experiment are grouped first by main group (Science/Ancillary), then in any by category (Moments, Particle Distribution, ...). If datasets with the same gui title have different units, a drop-down list will be displayed at the top of the category allowing to select the current unit.

7.12.2.3 Test pass-fail criteria

Test passed if the output matches the list of datasets in the appendix, failed otherwise.

7.12.2.4 Environmental needs

Not applicable.

7.12.2.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.12.3 Test case UI-SEARCH-03: Cancel search

This test validates that the progress bar and mouse correctly indicate that a search is in progress and that the Cancel button stops a search successfully.

7.12.3.1 Inputs specification

- 1- Perform a datasets search without criteria.
- 2- Click on “Cancel” button before results tab is displayed (“Data Request” tab).

7.12.3.2 Outputs specification

- 1- While the query search is being performed the progress bar and mouse correctly indicate that a task is in progress.



2- After clicking on “Cancel” button, the progress bar and mouse return to its idle status, and no results tab is displayed.

7.12.3.3 Test pass-fail criteria

Test passed if the progress bar and mouse correctly indicate progress and that cancelling a search stops the search successfully, failed otherwise

7.12.3.4 Environmental needs

Not applicable.

7.12.3.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.12.4 Test case UI-SEARCH-04: Search for all inventory data

This test validates that performing a search of all available inventory data returns the correct number of matching results.

7.12.4.1 Inputs specification

No criteria entered before performing the search on inventory.

7.12.4.2 Outputs specification

“Inventory Request” tab is displayed listing the expected inventory records.

7.12.4.3 Test pass-fail criteria

Test passed if the correct number of results is returned, failed otherwise.

7.12.4.4 Environmental needs

Not applicable.

7.12.4.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.12.5 Test case UI-SEARCH-05: Search datasets with a given criteria

This test validates that performing a datasets search with constraints in Date Range, Experiment, Measurement Type, Dataset Title, Dataset ID and/or main group and categories returns the correct number of matching results.

7.12.5.1 Inputs specification

Date Range: 2004-10-01 :00:00 / 2004-10-05 00:00:00.

Experiments: EDI and FGM.

Measurement Type: #####

Dataset ID: SPIN

7.12.5.2 Outputs specification

“Data Request” tab is displayed listing 2 experiments ‘EDI’ and ‘FGM’. Each experiment contains only 1 dataset title, available for the 4 s/c.

7.12.5.3 Test pass-fail criteria

Test passed if the correct number of relevant results is displayed, failed otherwise.

7.12.5.4 Environmental needs

Not applicable.

7.12.5.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.12.6 Test case UI-SEARCH-06: Clear search

This test validates that the Clear button successfully clears previous search criteria.

7.12.6.1 Inputs specification

- 1- In a search tab with criteria entered, click on “Clear” button.
- 2- Afterward perform a datasets search.

7.12.6.2 Outputs specification

- 1- Search criteria have been removed, text fields are empty and option lists have taken the default value (“All”).
- 2- A dataset search returns the same datasets as in Test Case UI-SEARCH-02.

7.12.6.3 Test pass-fail criteria

Test passed if the “Clear” button successfully clears previous search criteria, and next search return the expected output, failed otherwise.

7.12.6.4 Environmental needs

Not applicable.

7.12.6.5 Interface dependencies

Test case UI-SEARCH-05 must be executed before.



7.12.7 Test case UI-SEARCH-07: Search inventory data with a given criteria

This test validates that performing an inventory search with constraints in Date Range, Experiment, Measurement Type, Dataset Title, Dataset ID and/or main group and categories returns the correct number of matching results.

7.12.7.1 Inputs specification

Date Range: 2004-10-01 :00:00 / 2004-10-05 00:00:00.

Experiments: EDI and FGM.

Measurement Type: #####

Dataset ID: SPIN

7.12.7.2 Outputs specification

“Inventory Request” tab is displayed listing 154 results, between EDI and FGM inventory data.

7.12.7.3 Test pass-fail criteria

Test passed if the correct number of relevant results is displayed, failed otherwise.

7.12.7.4 Environmental needs

Not applicable.

7.12.7.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.12.8 Test case UI-SEARCH-08: Validation of input fields at Search

This test validates that for some input fields a validation check is done.

7.12.8.1 Inputs specification

1- Date Range: 2004-10-10 00:00:00 / 2004-10-05 00:00:00.

2- Clear

7.12.8.2 Outputs specification

1- At the status bar, the error message “Date Range and Duration. End Date must be after the begin date” is displayed. Only active button is “Clear”, i.e., no query can be performed.

2- Date Range field is emptied and the error message disappears from status bar.

7.12.8.3 Test pass-fail criteria

Test passed if the error message is displayed and “Inventory” and “Query” buttons are disabled, failed otherwise.

7.12.8.4 Environmental needs

Not applicable.

7.12.8.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.13 UI-RESULTS Test Cases

7.13.1 Test case UI-RESULTS-01: Short-list of datasets

This test validates that when “short-list” option is selected, only those “key” datasets are displayed. In addition, it validates that when the “short-list” checkbox is unchecked, the full list of datasets is displayed.

7.13.1.1 Inputs specification

- 1- Select checkbox “short-list” in the Search tab and query for all datasets.
- 2- Unselect checkbox “short-list” in the “Data Request” tab.
- 3- Select again checkbox “short-list” in the “Data Request” tab.

7.13.1.2 Outputs specification

- 1- “Data Request” tab is displayed with the “short-list” checkbox selected and the “key” datasets.
- 2- After unselecting the checkbox “short-list” the full list of datasets is displayed (check result with output from UI-SEARCH-02).
- 3- After selecting again the checkbox, the list of datasets is reduced to match the output after the first action.

7.13.1.3 Test pass-fail criteria

Test passed if after every action the datasets displayed matched the expected list, failed otherwise.

7.13.1.4 Environmental needs

Not applicable.

7.13.1.5 Interface dependencies

Either test case UI-BASICS-01, UI-BASICS-02, UI-BASICS-03, UI-BASICS-04 or UI-BASICS-05 must be executed before.

7.13.2 Test case UI-RESULTS-02: Display Results Details

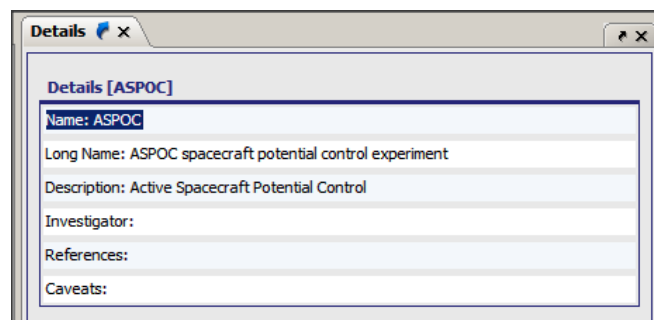
This test validates that selecting the magnifying glass of an Experiment opens a separate details panel containing the full list of metadata attributes of the experiment, and that they are displayed correctly. In addition, it validates that going over the Dataset Title, the Dataset Coverage and Dataset Id are displayed, and clicking on the Dataset Title opens a separate details panel containing the full list of dataset metadata attributes and that they are displayed correctly.

7.13.2.1 Inputs specification

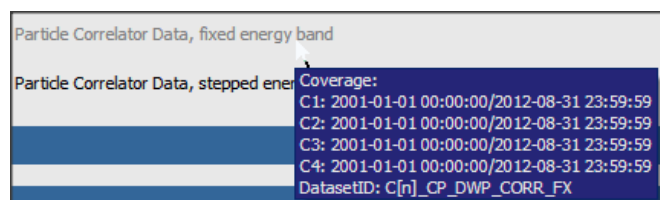
- 1- At “Data Request” tab the magnifying glass of the Experiment “ASPOC” is selected.
- 2- At “Data Request” tab the mouse go over the DWP dataset “Particle Correlator Data, fixed energy band” title.
- 3- At “Data Request” tab click on the title of dataset aforementioned.

7.13.2.2 Outputs specification

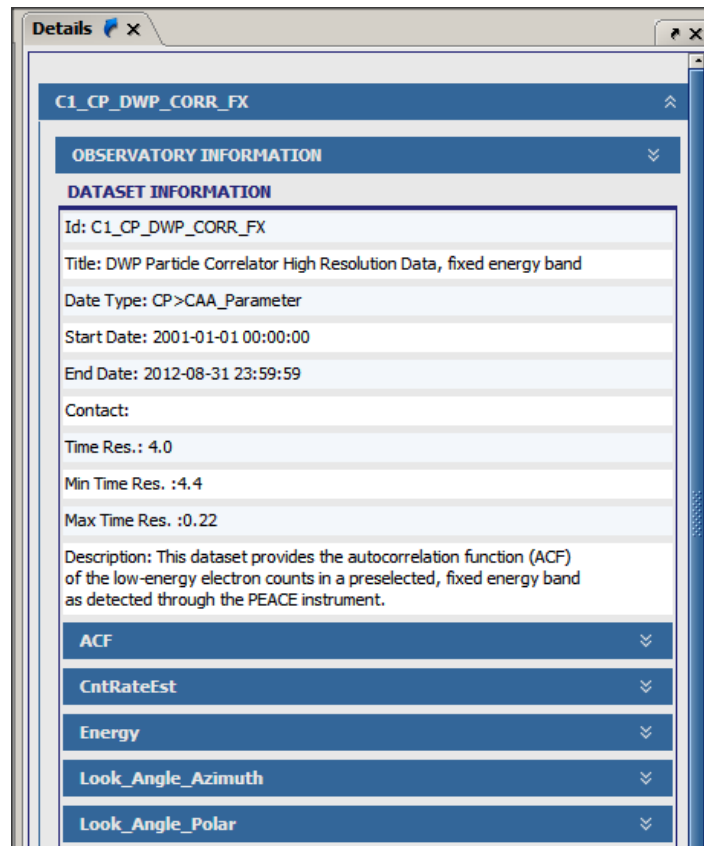
- 1- “Details” tab displaying ASPOC experiment metadata appears:



- 2- Floating panel with dataset coverage and Ids appears:



- 3- “Details” tab with datasets metadata appears:



7.13.2.3 Test pass-fail criteria

Test passed if the metadata attributes are displayed correctly in a separate details panel and floating panel, failed otherwise.

7.13.2.4 Environmental needs

Not applicable.

7.13.2.5 Interface dependencies

Test case UI-SEARCH-02 must be executed before.

7.13.3 Test case UI-RESULTS-03: Results pagination

This test validates that the results are displayed correctly in pages with the correct number of rows per page and that it is possible to move between pages using the arrows or directly entering a page number. This applies to all results panels (inventory and datasets). In addition, it validates that the columns of results can be sorted and the pagination value is kept.

7.13.3.1 Inputs specification

- 1- The 1st page of the 154 results from test case UI-SERACH-07 is displayed. Navigate to the previous/next pages.
- 2- Enter a certain page number at the text field.
- 3- Choose a column to order by its values (for example at inventory data: 'Dataset ID')

7.13.3.2 Outputs specification

- 1- Results are displayed in navigable pages.
When navigating to next page, next [n] results are displayed ([n] is page size).
When navigating to previous page, previous [n] results are displayed.
- 2- The results whose order is within [n]x[page_number] until ([n]x[page_number] + [n] - 1) are displayed.
- 3- Results are ordered by the column selected, but the page number did not change.

7.13.3.3 Test pass-fail criteria

Test passed if results are correctly displayed in navigable pages, failed otherwise.

7.13.3.4 Environmental needs

Not applicable.

7.13.3.5 Interface dependencies

Test case UI-SEARCH-07 must be executed before.

7.13.4 Test case UI-RESULTS-04: Results dynamic page size

This test validates that upon changing the page size, the results page updates dynamically and contains the appropriate maximum number of rows per page.

7.13.4.1 Inputs specification

The 1st page of the 154 results from test case UI-SERACH-07 is displayed with 25 results (default number of results per page).

- 1- Select a bigger number for records per page than the current one.
- 2- Select a smaller number for records per page than the current one.

7.13.4.2 Outputs specification

- 1- The number of records displayed at the current page matches the selected one, and the number of pages has decreased.
- 2- The number of records displayed at the current page matches the selected one, and the number of pages has increased.

7.13.4.3 Test pass-fail criteria

Test passed if changing the page size is successful, failed otherwise.



7.13.4.4 Environmental needs

Not applicable.

7.13.4.5 Interface dependencies

Test case UI-SEARCH-07 must be executed before.

7.13.5 Test case UI-RESULTS-05: Change Time constraint at Data Request tab

This test validates that upon changing the date range at the “Data Request” and refreshing; the “Data Request” page updates dynamically and contains the datasets available for the new time period.

7.13.5.1 Inputs specification

All datasets are displayed with a mission date range by default (at the time of writing this document = ‘2001-01-01 00:00:00 / 2014:12:31 23:59:59”).

Enter a new date range and refresh:

2004-10-01 00:00:00 / 2004-10-05 00:00:00

7.13.5.2 Outputs specification

A new query is performed and the number of datasets displayed is lower than before, because the datasets that do not contain data for that period are not displayed.

7.13.5.3 Test pass-fail criteria

Test passed if the datasets displayed are the ones that have records for that time, failed otherwise.

7.13.5.4 Environmental needs

Not applicable.

7.13.5.5 Interface dependencies

Test case UI-SEARCH-02 must be executed before.

7.13.6 Test case UI-RESULTS-06: Selection of dataset Units

This test validates that when a unit is selected at a drop-down units list contained in a certain category, all datasets belonging to that category and unit are displayed.

7.13.6.1 Inputs specification

Default display units for the “Particle Distribution” category of PEACE Ancillary datasets is “Particle Flux” (check by passing the mouse over any relevant title). Select a new unit: “count/s”



7.13.6.2 Outputs specification

The PEACE Ancillary Particle Distribution category displays two datasets more, and the Dataset IDs of the displayed datasets ends with 'cnts'.

7.13.6.3 Test pass-fail criteria

Test passed if the dataset IDs that belong to the given category have changed accordingly, failed otherwise.

7.13.6.4 Environmental needs

Not applicable.

7.13.6.5 Interface dependencies

Test case UI-SEARCH-02 must be executed before.

7.13.7 Test case UI-RESULTS-07: Clear datasets selection

This test validates that “Clear” button removes the selection of all selected datasets and disables the “Add to profile”, “Add to Shopping Basket” and “Direct Download” buttons.

7.13.7.1 Inputs specification

- 1- Select several datasets.
- 2- Clear selection.

7.13.7.2 Outputs specification

- 1- Dataset radio buttons are selected and “Add to profile”, “Add to Shopping Basket”, “Clear” and “Direct Download” buttons are enabled.
- 2- None of the radio buttons is selected anymore and “Add to profile”, “Add to Shopping Basket”, “Clear” and “Direct Download” buttons are disabled.

7.13.7.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.13.7.4 Environmental needs

Not applicable.

7.13.7.5 Special procedural constraints

Selection of datasets requires login and will also be tested from the security perspective later.

7.13.7.6 Interface dependencies

Test case UI-SEARCH-02 must be executed before.



7.13.8 Test case UI-RESULTS-08: Save datasets in a profile

This test validates that selected datasets can be added to an existing “General” profile or to a new one.

7.13.8.1 Inputs specification

1. Select several datasets and create a new Profile called “TC UI-Results-07”.
2. Clear selection. Select different datasets and add them to “TC UI-Results-07”.
3. Remove “TC UI-Results-07” profile.

7.13.8.2 Outputs specification

1. A General profile has been created with name “TC UI-Results-07” and lists the selected datasets.
2. The General profile “TC UI-Results-07” lists the previous datasets plus the new ones.
3. General profile “TC UI-Results-07” is not available anymore at the General profiles tab.

7.13.8.3 Test pass-fail criteria

Test passed if the dataset datasets are added to a General profile as expected, failed otherwise.

7.13.8.4 Environmental needs

Not applicable.

7.13.8.5 Special procedural constraints

Selection of datasets and profiles requires login and will also be tested from the security perspective later.

7.13.8.6 Interface dependencies

Test case UI-SEARCH-07 should have been passed successfully.
Test case UI-SEARCH-02 must be executed before.

7.13.9 Test case UI-RESULTS-09: Choose delivery format

This test validates that the delivery format for Direct Download can be selected at the “Data Request” tab.

7.13.9.1 Inputs specification

Select several datasets. Open the Delivery Format drop-down list.

7.13.9.2 Outputs specification

Drop-down list is displayed with options: “CEF”, “CDF” and “CDF_2_7”.



7.13.9.3 Test pass-fail criteria

Test passed if the delivery format drop-down list is working, failed otherwise.

7.13.9.4 Environmental needs

Not applicable.

7.13.9.5 Special procedural constraints

Selection of datasets requires login and will also be tested from the security perspective later.

7.13.9.6 Interface dependencies

Test case UI-SEARCH-02 must be executed before.

7.13.10 Test case UI-RESULTS-10: Add results to Shopping Basket

This test validates that selected datasets can be sent to the Shopping Basket successfully and, in doing so, checks whether the appropriate items are displayed correctly in the Shopping Basket overview window.

7.13.10.1 Inputs specification

1. Select several datasets and add them to Shopping Basket.
2. Open Shopping Basket tab.
3. At the Shopping Basket, perform a check-out.

7.13.10.2 Outputs specification

1. Shopping Basket Overview window appears with the selecting datasets displaying Dataset Id and current time range selected (note that if time range is modified and the datasets are re-added a new record with the Dataset id and the time range will be added to this window).
2. Shopping Basket tab is displayed with all datasets selected (note that are the same as in Shopping Basket Overview). The CEF Datasets in addition to the time range, have a drop-down list that allows to select the delivery format and a drop-down list that allows to select the interval of the delivered files.
3. The datasets are successfully removed from the shopping Basket.

7.13.10.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.13.10.4 Environmental needs

Not applicable.

7.13.10.5 Special procedural constraints

Selection of datasets and shopping basket requires login and will also be tested from the security perspective later.

7.13.10.6 Interface dependencies

Test case UI-SEARCH-05 must be executed before.

7.13.11 Test case UI-RESULTS-11: Remove results from Shopping Basket

This test validates whether Shopping Basket items can be removed manually in several ways, either individually, or in a group or altogether.

7.13.11.1 Inputs specification

1. Select an item (dataset) and remove it individually by clicking on the item's shopping cart icon with a minus.
2. Select several items and remove them by clicking on the shopping cart icon with a minus at the top of the table.
3. Use the "Remove All" button at the bottom of the tab.

7.13.11.2 Outputs specification

1. The removed item disappears from the list.
2. The removed group of items disappear from the list.
3. A confirmation dialog box appears, and when confirmed, all items are removed and the Shopping Basket tab is closed.

7.13.11.3 Test pass-fail criteria

1. Test passed if items can be removed from the Shopping Basket as described above successfully. Also tested is that Shopping Basket items are removed automatically when checked-out.
2. The entries in the Shopping Basket should be reflected correctly in the Shopping Basket Overview window.
3. Failed otherwise.

7.13.11.4 Environmental needs

Not applicable.

7.13.11.5 Special procedural constraints

Selection of datasets and shopping basket requires login and will also be tested from the security perspective later.

7.13.11.6 Interface dependencies

Test case UI-SEARCH-10 must be executed before.



7.13.12 Test case UI-RESULTS-12: Docking/Undocking Details windows

This test validates that the docking and undocking of details windows functions correctly.

7.13.12.1 Inputs specification

1. Undock the details window.
2. Dock again the details window.

7.13.12.2 Outputs specification

1. The Details window is removed from the main panel and displayed as an isolated window.
2. The isolated Details window is closed and it is displayed within the main window again.

7.13.12.3 Test pass-fail criteria

Test passed if the Details window can be docked and undocked successfully, failed otherwise.

7.13.12.4 Environmental needs

Not applicable.

7.13.12.5 Interface dependencies

Test case UI-RESULTS-02 must be executed before.

7.13.13 Test case UI-RESULTS-13: Docking/Undocking Results tabs

This test validates that the Results tab can be undocked and re-docked successfully.

7.13.13.1 Inputs specification

1. Undock the Results tab.
2. Dock again the Results tab.

7.13.13.2 Outputs specification

1. The Results tab is removed from the main panel and displayed as an isolated window.
2. The isolated Results window is closed and it is displayed within the main panel again.

7.13.13.3 Test pass-fail criteria

Test passed if results tab is successfully docked and undocked, failed otherwise.

7.13.13.4 Environmental needs

Not applicable.

7.13.13.5 Interface dependencies

Test case UI-SEARCH-02 must be executed before.

7.14 UI-RETRIEVAL Test Cases

7.14.1 Test case *UI-RETRIEVAL-01: Direct Download of Cluster files*

This test validates that files of CEF format from one or several datasets can be directly downloaded. As well as testing that for all CEF files the delivery format and interval can be chosen.

7.14.1.1 Inputs specification

1. Select several datasets that have CEF files. Choose CEF as delivery format. Click on “Download” button and select “All” as delivery interval.
2. Select several datasets that have CEF files. Choose CDF as delivery format. Click on “Download” button and select “1 day” as delivery interval.
3. Change time range to 1 hour. Select several datasets that have CEF files. Choose CDF_2_7 as delivery format. Click on “Download” button and select “10 minutes” as delivery interval.

7.14.1.2 Outputs specification

1. Once size to download has been confirmed, the progress bar and mouse change to on-going task mode, and after a while a save dialog window is displayed to save the result in a gzip file. The result file contains one file per dataset with the requested data in CEF format.
2. Once size to download has been confirmed, the progress bar and mouse change to on-going task mode, and after a while a save dialog window is displayed to save the result in a gzip file. The result file contains several files per dataset with the requested data in CDF format. Each file covers 1 day.
3. Once size to download has been confirmed, the progress bar and mouse change to on-going task mode, and after a while a save dialog window is displayed to save the result in a gzip file. The result file contains several files per dataset with the requested data in CDF 2.7 format. Each file covers 10 minutes.

7.14.1.3 Test pass-fail criteria

Test passed if a result file can be saved locally to disk fulfilling the output expected, failed otherwise.

7.14.1.4 Environmental needs

Software able to uncompress and untar tar.gz files is required (Example: “gunzip” and “tar xvf ..”).

7.14.1.5 Special procedural constraints

Selection of datasets and download require login and will also be tested from the security perspective later.



7.14.1.6 Interface dependencies

Test case UI-SEARCH-05 must be executed before.

7.14.2 Test case UI-RETRIEVAL-02: Direct Download of non-Cluster files

This test validates that files of non-CEF format from one or several datasets can be directly downloaded.

7.14.2.1 Inputs specification

Change time range to 1 hour. Select several datasets that have non-CEF files (for example, from WBD experiment). Click on “Download” button and select “All” as delivery interval.

Date Range: 2004-10-03 09:00:00 / 2004-10-03 10:00:00

Note that Delivery Format and Delivery Interval have no effect over non-Cluster file requests.

7.14.2.2 Outputs specification

Once size to download has been confirmed, the progress bar and mouse change to on-going task mode, and after a while a save dialog window is displayed to save the result in a gzip file. The result file contains all the original files of the selected datasets whose time coverage overlaps with the time range selected.

(For example, if dataset “WBD Survey Spectrograms” was selected, three files are returned:

“CM_CG_WBD_SPEC_PLOT_OVERVIEW_PNG__20041003_0819_20041003_0947_V00.png”,

“CM_CG_WBD_SPEC_PLOT_OVERVIEW_PNG__20041003_0820_20041003_0948_V00.png”

and

“CM_CG_WBD_SPEC_PLOT_OVERVIEW_PNG__20041003_0821_20041003_0947_V00.png”)

7.14.2.3 Test pass-fail criteria

Test passed if a result file can be saved locally to disk fulfilling the output expected, failed otherwise.

7.14.2.4 Environmental needs

Software able to uncompress and untar tar.gz files is required (Example: “gunzip” and “tar xvf ..”).

7.14.2.5 Special procedural constraints

Selection of datasets and download require login and will also be tested from the security perspective later.

7.14.2.6 Interface dependencies

Test case UI-SEARCH-02 must be executed before.

7.14.3 Test case UI-RETRIEVAL-03: Download of Cluster files from Shopping Basket

This test validates that files of CEF format from one or several datasets sent to Shopping Basket can be downloaded asynchronously. As well as testing that for all CEF files the delivery format and interval can be chosen.

7.14.3.1 Inputs specification

1. Select several datasets that have CEF files. Choose different delivery format per item. Choose different delivery interval per item. Click on “Check Out” button.
2. Follow the ftp link included in the e-mail received.

7.14.3.2 Outputs specification

1. An information message appears with the request ID and the email address where the processed information e-mail will be sent to.
When the data is ready to download, an e-mail arrived to the address aforementioned. The e-mail contains an ftp link where the user can download the data from.
2. A dialog box appears to save the result in a gzip file. The result file contains the file(s) per dataset with the requested data in the requested format. Each file covers the delivery interval selected.

7.14.3.3 Test pass-fail criteria

Test passed if the requested data is downloaded asynchronously as expected, failed otherwise.

7.14.3.4 Environmental needs

Software able to uncompress and untar tar.gz files is required (Example: “gunzip” and “tar xvf ..”).

7.14.3.5 Special procedural constraints

Selection of datasets and Shopping Basket require login and will also be tested from the security perspective later.

7.14.3.6 Interface dependencies

Test case UI-RESULTS-10exitI guess must be executed before.

7.14.4 Test case UI-RETRIEVAL-04: Download of non-Cluster files from Shopping Basket

This test validates that files of non-CEF format from one or several datasets sent to Shopping Basket can be downloaded asynchronously.

7.14.4.1 Inputs specification

1. Select several datasets that have non-CEF files. Click on “Check Out” button.



2. Follow the ftp link included in the e-mail received.

7.14.4.2 Outputs specification

1. An information message appears with the request ID and the email address where the processed information e-mail will be sent to.
When the data is ready to download, an e-mail arrived to the address aforementioned. The e-mail contains an ftp link where the user can download the data from.
2. A dialog box appears to save the result in a gzip file. The result file contains all the original files of the selected datasets whose time coverage overlaps with the time range selected

7.14.4.3 Test pass-fail criteria

Test passed if the requested data is downloaded asynchronously as expected, failed otherwise.

7.14.4.4 Environmental needs

Software able to uncompress and untar tar.gz files is required (Example: “gunzip” and “tar xvf ..”).

7.14.4.5 Special procedural constraints

Selection of datasets and Shopping Basket require login and will also be tested from the security perspective later.

7.14.4.6 Interface dependencies

Test case UI-RESULTS-10exitI guess must be executed before.

7.15 UI-PLOTTING Test Cases

7.15.1 Test case UI-PLOTTING-01: Selection of time interval

This test validates that for the Date Range input fields a validation check is done.

7.15.1.1 Inputs specification

Introduce a wrong Date Range. For ex.: 2004-10-10 :00:00 / 2004-10-05 00:00:00.

7.15.1.2 Outputs specification

At the status bar the error message “Date Range and Duration. End Date must be after the begin date” is displayed. “Plot Panel(s)”, “Previous Plot” and “Next Plot” are disabled, i.e., no plotting can be performed.

7.15.1.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.15.1.4 Environmental needs

Not applicable.

7.15.1.5 Special procedural constraints

Not applicable.

7.15.1.6 Interface dependencies

Test case UI-BASICS-06 must be executed before using the option of “Graphical Products”.

7.15.2 Test case UI-PLOTTING-02: Selection of Plot Type

This test validates that different plot types can be selected. In addition, it validates that when a “Pre-generated” type is selected end time and duration at Date Range cannot be modified by the user, and that end time is equals to the start time plus the duration of the pre-generated type selected.

7.15.2.1 Inputs specification

- 1- Introduce a Date Range, for ex.: 2004-10-05 23:45:24 / 2004-10-07 00:00:00 and select “on-demand” type.
- 2- Select “Pre-generated 1 Hour”.
- 3- Select “on-demand” type, reintroduce date range of step 1 and select “Pre-generated 6 hours”.
- 4- Select “on-demand” type, reintroduce date range of step 1 and select “Pre-generated 1 Day”.

7.15.2.2 Outputs specification

- 1- Date Range fields do not change (any duration is allowed).
- 2- “End time” and “Duration” become non-editable. “Start” changes to “2004-10-05 23:00:00”, “End” changes to “2004-10-06 00:00:00” and “Duration” will be “1 hour”.
- 3- “End time” and “Duration” become non-editable. “Start” changes to “2004-10-05 23:00:00”, “End” changes to “2004-10-06 05:00:00” and “Duration” will be “6 hours”.
- 4- “End time” and “Duration” become non-editable. “Start” changes to “2004-10-05 23:00:00”, “End” changes to “2004-10-06 23:00:00” and “Duration” will be “1 Day”.

7.15.2.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.15.2.4 Environmental needs

Not applicable.

7.15.2.5 Special procedural constraints

Not applicable.

7.15.2.6 Interface dependencies

Test case UI-BASICS-06 must be executed before using the option of “Graphical Products”.

7.15.3 Test case UI-PLOTTING-03: Selection of Graphical Products and sorting

This test validates that more than one Graphical product can be selected to be embedded in the same plot (note that every Graphical product is plotted in a panel, and those panels will be stacked together to create the image to display). In addition, it validates that the order in which the panels (Graphical products) will be stacked can be modified.

7.15.3.1 Inputs specification

- 1- Select several Graphical products by selecting the corresponding radio-buttons.
- 2- Select one Graphical product identifier from the bottom of the list of panels and move it upwards.
- 3- Select one Graphical product identifier from the top of the list of panels and move it downwards.

7.15.3.2 Outputs specification

- 1- The identifier of every Graphical product is added to the list of panels in the text area at the top-right corner of the window. The identifier corresponds to the 3 first letters of the experiment, plus the observatory number, followed by the gui title of the Graphical product.
- 2- The identifier moves up in the list.
- 3- The identifier moves down in the list.

7.15.3.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.15.3.4 Environmental needs

Not applicable.

7.15.3.5 Special procedural constraints

Not applicable.

7.15.3.6 Interface dependencies

Test case UI-BASICS-06 must be executed before using the option of “Graphical Products”.

7.15.4 Test case UI-PLOTTING-04: Display Plot Result

This test validates that selecting to plot one or several graphical products opens a separate plots panel containing the results of the plot action, i.e. an image, and that they are displayed correctly.



7.15.4.1 Inputs specification

Enter a time criteria, select a plot type and select one or more Graphical Products. Click on “Plot panel(s)” button.

7.15.4.2 Outputs specification

“Plot” tab displaying the image with the results returned is displayed in a new panel at the right side of the main panel.

7.15.4.3 Test pass-fail criteria

Test passed if the plot results are displayed correctly in a separate plots panel, failed otherwise.

7.15.4.4 Environmental needs

Not applicable.

7.15.4.5 Special procedural constraints

Perform a plot requires login and will also be tested from the security perspective later

7.15.4.6 Interface dependencies

Test case GRAPHICS-BASIC-01 and GRAPHICS-BASIC-02 are passed.

Test case UI-BASICS-06 must be executed before using the option of “Graphical Products”.

7.15.5 Test case UI-PLOTTING-05: Save Graphical products in a Graphical profile

This test validates that selected Graphical products can be added to an existing “Graphical” profile or to a new one.

7.15.5.1 Inputs specification

- 1- Select several Graphical products and create a new Profile called “TC UI-Plotting-05”.
- 2- Clear selection. Select different datasets and add them to “TC UI-Plotting-05”.
- 3- Remove “TC UI-Plotting-05” profile

7.15.5.2 Outputs specification

- 1- A Graphical profile has been created with name “TC UI-Plotting-05” and lists the selected Graphical products.
- 2- The Graphical profile “TC UI-Plotting-05” lists the previous Graphical products plus the new ones.
- 3- Graphical profile “TC UI-Plotting-05” is not available anymore at the Graphical profiles tab.



7.15.5.3 Test pass-fail criteria

Test passed if the Graphical products are added to a Graphical profile as expected, failed otherwise.

7.15.5.4 Environmental needs

Not applicable.

7.15.5.5 Special procedural constraints

Perform a plot requires login and will also be tested from the security perspective later.

7.15.5.6 Interface dependencies

Test case UI-PLOTTING-03 may be used to start this test case.

7.15.6 Test case UI-PLOTTING-06: Plot with no data returned

This test validates that, when searching using criteria for which there are no results, the application will display an empty plot.

7.15.6.1 Inputs specification

Select a Graphical product and perform a plotting with criteria for which there are not results. Ex.: ASPOC – C4, 2001-11-08 and 1 Day.

7.15.6.2 Outputs specification

Progress bar and mouse indicates that a task is in progress. Once the results are returned, an empty plot is displayed.

7.15.6.3 Test pass-fail criteria

Test passed if the output is displayed, failed otherwise.

7.15.6.4 Environmental needs

Not applicable.

7.15.6.5 Special procedural constraints

Perform a plot requires login and will also be tested from the security perspective later

7.15.6.6 Interface dependencies

At least test cases GRAPHICS-BASIC-01 is passed.

Test case UI-BASICS-06 must be executed before using the option of “Graphical Products”.

7.15.7 Test case UI-PLOTTING-07: Stack of plots and navigation

This test validates that, when more than one Graphical product is selected, the result panels per Graphical product are stacked one on top of the other following the order defined at the list of panels. In addition, it validates that for on-demand plots, the maximum number of stacked panels per page is 7, whenever there are more than 7 panels the leftover are displayed in another page. Furthermore, this test validates that there is a way to navigate forward and backward when several pages are returned, as well as, it is possible to navigate through all the plots obtained until now within the current session.

7.15.7.1 Inputs specification

- 1- Select 9 Graphical products and perform an on-demand plot for a certain Date Range (for example 5 minutes).
- 2- At “Plots” panel, click on “Next plot” arrow.
- 3- At “Plots” panel, click on “Previous plot” arrow.
- 4- Perform any other plot with different type or Graphical product.
- 5- At “Plots panel”, use the arrows to navigate through the returned images.

7.15.7.2 Outputs specification

- 1- Progress bar and mouse indicates that a task is in progress. Once the results are returned, the first page with the panels of the Graphical products ordered as specified at the list of panels is displayed.
- 2- The second page with the remaining 2 panels is displayed.
- 3- The first page is displayed again.
- 4- The resulting plot is displayed.
- 5- All the images obtained until now can be reached by using the arrow buttons at “Plots” panel.

7.15.7.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.15.7.4 Environmental needs

Not applicable.

7.15.7.5 Special procedural constraints

Perform a plot requires login and will also be tested from the security perspective later

7.15.7.6 Interface dependencies

At least test case GRAPHICS-BASIC-01 is passed.

Test case UI-BASICS-06 must be executed before using the option of “Graphical Products”.

7.15.8 Test case UI-PLOTTING-o8: Plots next/previous duration

This test validates that, when a plot has been performed for a set of Graphical products for a given Date Range, it is possible to perform the same plot with the same duration for the previous time range and for the next time range just with one click.

7.15.8.1 Inputs specification

- 1- Select one or more Graphical products and perform an on-demand plot for a certain Date Range (for example 5 minutes).
- 2- At “Graphical Products” panel, click on “Next plot in time” arrow.
- 3- Re-enter the Date range criteria, plot it, and at “Graphical Products” panel, click on “Previous plot in time” arrow.
- 4- Repeat operation with pre-generated plots, for example “Pre-generated 1 Hour”.

7.15.8.2 Outputs specification

- 1- Progress bar and mouse indicates that a task is in progress. Once the results are returned, the plot corresponding to the Date range selected is displayed.
- 2- Date Range is updated to start time + duration, and end time + duration. Plotting task with the new date range starts. Progress bar and mouse indicates that a task is in progress. Once the results are returned, the plot corresponding to the new Date range is displayed.
- 3- Date Range is updated to start time - duration, and end time - duration. Plotting task with the new date range starts. Progress bar and mouse indicates that a task is in progress. Once the results are returned, the plot corresponding to the new Date range is displayed.
- 4- When right arrow, “Next”, is clicked, the plot returned corresponds to the next hour, when left arrow, “Previous”, is clicked, the plot returned corresponds to the previous hour.

7.15.8.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.15.8.4 Environmental needs

Not applicable.

7.15.8.5 Special procedural constraints

Perform a plot requires login and will also be tested from the security perspective later

7.15.8.6 Interface dependencies

Test case GRAPHICS-BASIC-01 and GRAPHICS-BASIC-02 are passed.

Test case UI-BASICS-06 must be executed before using the option of “Graphical Products”.

7.15.9 Test case UI-PLOTTING-09: Docking/Undocking Plots windows

This test validates that the docking and undocking of plots windows functions correctly, as well as the detach functions. This test validates, also, the display of the plot images as a Postcard and the functions associated. In addition, it validates when the plots window is closed, it can be opened from the Windows menu displaying the same content as before closing.

7.15.9.1 Inputs specification

- 1- Undock the Plots window.
- 2- Dock again the Plots window.
- 3- Detach (blue arrow at the top of the tab) the Plots window.
- 4- Execute menu item “Plots” from menu “Windows”.
- 5- Click on the plot image.
- 6- Detach the Postcard window.

7.15.9.2 Outputs specification

- 1- The Plots window is removed from the main panel and displayed as an isolated window.
- 2- The isolated Plots window is closed and it is displayed within the main window again.
- 3- The Plots window is removed from the main panel and displayed as an isolated window, but the content is frozen, that is, whenever new plots are performed, they are not added to the sequence of images for navigation.
- 4- The Plots window is displayed within the main window at the right panel (as by default).
- 5- A floating Postcard window is displayed showing a copy of the content of the Plots window. Whenever new plots are performed, they will be added to the sequence of images for navigation.
- 6- The Postcard window is closed, and a Detached Postcard window is displayed instead, but the content is frozen.

7.15.9.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.15.9.4 Environmental needs

Not applicable.

7.15.9.5 Interface dependencies

Either test case UI-PLOTTING-07 or UI-PLOTTING-08 must be executed before.

7.15.10 Test case UI-PLOTTING-10: Retrieve plot in Postscript format

This test validates that when a plotting has been performed, the result can be retrieved and saved in local disk in Postscript format.



7.15.10.1 Inputs specification

At “Plots” window, click on “PS” button.

7.15.10.2 Outputs specification

A save dialog window is displayed to save the result in Postscript format in the local disk.

7.15.10.3 Test pass-fail criteria

Test passed if a result file can be saved locally to disk fulfilling the output expected, failed otherwise.

7.15.10.4 Environmental needs

Not applicable.

7.15.10.5 Special procedural constraints

Perform a plot requires login and will also be tested from the security perspective later.

7.15.10.6 Interface dependencies

Test case UI-PLOTTING-07 or UI-PLOTTING-08 may be used to start this test case.

7.15.11 Test case UI-PLOTTING-11: Retrieve CEF source files

This test validates that when an “on-demand” plotting has been performed, the CEF source files used to generate the plot can be retrieved and saved locally in disk.

7.15.11.1 Inputs specification

At “Plots” window, click on “CEF” button.

7.15.11.2 Outputs specification

A save dialog window is displayed to save the source files compress in a zip file in the local disk.

7.15.11.3 Test pass-fail criteria

Test passed if a result file can be saved locally to disk fulfilling the output expected, failed otherwise.

7.15.11.4 Environmental needs

Software to uncompress zip files is needed.

7.15.11.5 Special procedural constraints

Perform a plot requires login and will also be tested from the security perspective later.



7.15.11.6 Interface dependencies

Test case UI-PLOTTING-07 or UI-PLOTTING-08 may be used to start this test case.

7.15.12 Test case UI-PLOTTING-12: Retrieve plot file in PNG format

This test validates that when a “pre-generated” plot has been requested, the returned image can be saved locally in PNG format.

7.15.12.1 Inputs specification

At “Plots” window, click on “PNG” button.

7.15.12.2 Outputs specification

A save dialog window is displayed to save the image file in png format in the local disk.

7.15.12.3 Test pass-fail criteria

Test passed if a result file can be saved locally to disk fulfilling the output expected, failed otherwise.

7.15.12.4 Environmental needs

Not applicable.

7.15.12.5 Special procedural constraints

Perform a plot requires login and will also be tested from the security perspective later.

7.15.12.6 Interface dependencies

Test case UI-PLOTTING-07 or UI-PLOTTING-08 may be used to start this test case.

7.16 UI-SECURITY Test Cases

7.16.1 Test case UI-SECURITY-01: Log in/ logout

This test validates that the user can be authenticated against the RSSD LDAP system and that the login/logout functionality work successfully.

7.16.1.1 Inputs specification

- 1- Log in the CSA using RSSD LDAP credentials (username and password).
- 2- Logout.

7.16.1.2 Outputs specification

- 1- Status bar displays a message with username and the time of log in. At the menu bar the username replaces the text fields for Username and Password. The profiles button is enabled at the Tool bar.
- 2- Status bar displays message “Not Logged In”. At the menu bar the username has been replaced by the text fields Username and Password. The profiles button is disabled at the Tool bar.

7.16.1.3 Test pass-fail criteria

Test passed if the user is authenticated against the RSSD LDAP system correctly, failed otherwise.

7.16.1.4 Environmental needs

Not applicable.

7.16.1.5 Special procedural constraints

User need to be registered at the RSSD LDAP registry.

7.16.1.6 Interface dependencies

None.

7.16.2 Test case *UI-SECURITY-02: Register as a New User*

This test validates that selecting the Register menu item from the menu bar opens the Cluster User Registration page in a browser window. Similarly the page should be opened when the Sign Up button is clicked in the Login dialog window.

7.16.2.1 Inputs specification

Perform the menu action “Register as New User”.

7.16.2.2 Outputs specification

A browser window is opened at the Registration page of the Cluster Science Archive.

7.16.2.3 Test pass-fail criteria

Test passed if the registration page on the RSSD LDAP system opens in a browser window, failed otherwise.

7.16.2.4 Environmental needs

Not applicable.

7.16.2.5 Special procedural constraints

User need to be registered at the RSSD LDAP registry.



7.16.2.6 Interface dependencies

None.

7.16.3 Test case UI-SECURITY-03: Access to public services

This test validates that a user who has not logged in only has access to search for inventory data, search for datasets, and search for graphical products.

7.16.3.1 Inputs specification

- 1- Perform searches on inventory, datasets and graphical products.
- 2- Try to plot a graphical product or to select a dataset.

7.16.3.2 Outputs specification

- 1- “Profiles” button is disabled. Searches are performed successfully.
- 2- The authentication dialog box is displayed, requesting to enter RSSD LDAP credentials.

7.16.3.3 Test pass-fail criteria

Test passed if logged out user only has accessed to public data, failed otherwise.

7.16.3.4 Environmental needs

Not applicable.

7.16.3.5 Special procedural constraints

This test requires that the user is not logged in.

7.16.3.6 Interface dependencies

None.

7.16.4 Test case UI-SECURITY-04: Access to non-public services

This test validates whether the user, that is not already logged in, is requested to login when attempting to perform any non-public task. These tasks include: selection of datasets, direct download requests, shopping basket requests; access to profiles, plot of graphical products.

7.16.4.1 Inputs specification

- 1- At “Data Request” tab, select a dataset.
- 2- At “Graphical Products Search” tab, select one or several graphical products and click on “Plot Panels” window.

7.16.4.2 Outputs specification

1- “Profiles” button at Toolbar is disabled. “Add to Shopping Basket” and “Download” buttons are disabled at “Data Request” tab.

In both cases a dialog box requesting to log in is displayed.

7.16.4.3 Test pass-fail criteria

Test passed if the user is requested to log in when attempting to perform any non-public task, failed otherwise.

7.16.4.4 Environmental needs

Not applicable.

7.16.4.5 Special procedural constraints

This test requires that the user is not logged in.

7.16.4.6 Interface dependencies

Test case UI-SEARCH-05 must be executed.

7.17 UI-LOGGING Test Cases

7.17.1 Test case UI-LOGGING-01: Log Console

This test validates whether the Log Console correctly displays the logging output as actions are performed in the user interface.

7.17.1.1 Inputs specification

Make the Log Console window visible.

7.17.1.2 Outputs specification

After every action a log is displayed within the Log Console adding information about the action or providing error messages if the action failed.

7.17.1.3 Test pass-fail criteria

Test passed if the Log Console correctly displays the logging output, failed otherwise.

7.17.1.4 Environmental needs

Not applicable.

7.17.1.5 Interface dependencies

None.



7.17.2 Test case UI-LOGGING-02: Logging levels

This test validates whether changing the logging level correctly changes the detail of logs provided.

7.17.2.1 Inputs specification

Change the log level at the `csa.jnlp` file from INFO to DEBUG and open the UI.

7.17.2.2 Outputs specification

More detailed logging is performed.

7.17.2.3 Test pass-fail criteria

Test passed if detail of logging output changes when different levels are specified, failed otherwise.

7.17.2.4 Environmental needs

Not applicable.

7.17.2.5 Interface dependencies

None.

7.18 UI-PERFORMANCE Test Cases

Almost all performance tests can be run in a similar way thanks to the ANT target *testPerformance* added to the SVN repository for CSA. Before running the tests, some configuration parameters must be tuned. These parameters are commented out in the *build.properties.defaults* file for the UI application in the client layer. The parameters in the general section and the one referred to performance testing must be defined for instance in another file called *build.properties* (which is ignored in the SVN repository configuration for that application). These parameters are:

- *build.dir*=build. It must be defined like this whenever the rest of parameters are taken out to other file (as recommended).
- *grinder.console.host*=satl22.net4.lan. It specifies the host that will start the Grinder console where all the agents will connect and report the results. It is advised to use the Fully Qualified Domain Name where the tests are going to be run from.
- *grinder.jar.file*=/home/dtapiador/software/grinder-3.2/lib/grinder.jar. It specifies the Grinder jar containing the Grinder application.
- *grinder.runs*=0. It defines the number of times each test will be executed (0 means infinite). The value recommended here is 100 as shown in the test cases design.
- *conf.performance.dir*=test/performance. This parameter configures the location of the different script templates, etc. needed to run the tests. It should be left like it is.
- *build.performance.dir*=\${build.dir}/performance. It simply specifies where the tests should be run. It should be left as it is.



- *performance.tests*="NameOfClass1:testMethod1 NameOfClass2:testMethod2". This parameter is the most important one because it specifies the JUnit tests that will be run. The syntax is very intuitive and simply states a space separated test where each test consists of the class containing the test method (usually the one extending JUnit's TestCase) and the test method itself (the name as coded in the sources). When Grinder starts up, it will go one by one running these tests at the user commands.

The procedure for running these tests is very easy and can be explained with just a view of the user interface of Grinder's console. It represents the results view of the console, where all results are gathered and shown to the tester in a friendly format. The tester only has to start the processes for one test with the *Start processes* menu item in *Action* menu, and once all test instances (or whenever the processes are to be interrupted) have been executed (check *Successful Tests* plus *Errors* columns in the view and compare it to *grinder.runs* parameter), he/she can proceed with the next one by clicking on *Stop processes*, waiting a bit until next agents are ready and click again on *Start processes*. Agents are ready when there is a message in the terminal where *ant testPerformance* was run, saying that the agent is waiting for console signal. Whenever the tester wants to re-run the test, he/she must start again the processes when they have finished without clicking first on *Stop processes*.

See Validation test platforms requirements for further information on the Grinder java load testing framework.

The tests included in the design are very similar, so some of them will not be specified one by one because they can be run easily with Grinder tool. However, for certain tests, the test case procedure must be specified as it cannot be run with the Grinder (or it can, but just partially) (UI-PERFORMANCE-01, UI-PERFORMANCE-05, UI-PERFORMANCE-06, UI-PERFORMANCE-09).

7.18.1 Test case UI-PERFORMANCE-01: Web interface initialisation

This test validates whether the web interface initialises correctly and appropriately.

7.18.1.1 Inputs specification

Open Cluster Science Archive Java web start application.

7.18.1.2 Outputs specification

The main window is displayed with all the components readable.

7.18.1.3 Test pass-fail criteria

Test passed if the web interface initialises correctly and appropriately, failed otherwise.

7.18.1.4 Environmental needs

Not applicable.



7.18.1.5 Interface dependencies

None.

7.18.2 Test case UI-PERFORMANCE-05: Retrieval of CEF product from Shopping-Basket

This test checks how much time passes between the user requesting the retrieval of a CEF data file from the Shopping Basket and when he actually receives the mail with the URL for its download.

7.18.2.1 Inputs specification

Select a CEF dataset to download for a certain time range in a single file (“All” as delivery interval).

Grinder must be properly configured. It is recommended to run this test isolated from other tests and with one repetition only (*grinder.runs* parameter), as a greater number of tests would be unaffordable. The test therefore is started from Grinder.

7.18.2.2 Outputs specification

When data is ready to download an e-mail arrives to the user’s e-mail address. The time will be recorded.

7.18.2.3 Test pass-fail criteria

Test passed if the user receives an email with the correct URL in a time similar to the time obtained from retrieving the same data from CAA, failed otherwise.

7.18.2.4 Environmental needs

Not applicable.

7.18.2.5 Special procedural constraints

This test requires that the user is logged in.

7.18.2.6 Interface dependencies

Test UI-RETRIEVAL-03 can be used to select data to download.

7.18.3 Test case UI-PERFORMANCE-06: Retrieval of non-CEF product from Shopping-Basket

This test checks how much time passes between the user requesting the retrieval of a non-CEF data from the Shopping Basket and when he actually receives the mail with the URL for its download.



7.18.3.1 Inputs specification

Select a non-CEF dataset to download for a certain time range.

Grinder must be properly configured. It is recommended to run this test isolated from other tests and with one repetition only (*grinder.runs* parameter), as a greater number of tests would be unaffordable. The test therefore is started from Grinder.

7.18.3.2 Outputs specification

When data is ready to download an e-mail arrives to the user's e-mail address. The time will be recorded.

7.18.3.3 Test pass-fail criteria

Test passed if the user receives an email with the correct URL in a time similar to the time obtained from retrieving the same data from CAA, failed otherwise.

7.18.3.4 Environmental needs

Not applicable.

7.18.3.5 Special procedural constraints

This test requires that the user is logged in.

7.18.3.6 Interface dependencies

Test UI-RETRIEVAL-04 can be used to select data to download.

7.18.4 Test case UI-PERFORMANCE-09: Availability of requested files

This test validates that any retrieval request is available for download for one week after the request was made.

7.18.4.1 Inputs specification

Select a dataset(s) to download for a certain time range at the Shopping Basket. When data is ready to download an e-mail arrives to the user's e-mail address. The products will be downloaded one week (or rather, a couple of hours before) to check their availability.

7.18.4.2 Outputs specification

The data can be successfully saved to a local disk.

7.18.4.3 Test pass-fail criteria

Test passed if the data products are available to download one week after the request, failed otherwise.

7.18.4.4 Environmental needs

Not applicable.



7.18.4.5 Special procedural constraints

This test requires that the user is logged in.

7.18.4.6 Interface dependencies

Test Case UI-RETRIEVAL-03 or UI-RETRIEVAL-04 can be used to select data to download. But the link provided in the e-mail should not be followed until 1 week has passed.

7.19 UI-SCALABILITY Test Cases

In all test cases from this section, the cause of the poor throughput (once the performance requirements are not met) will need to be specified or identified (memory goes beyond the limit established, CPU is overloaded, Input/Output bottleneck, maximum request handlers reached, etc) whenever possible.

The **scalability analysis** that is to be performed in every feature to be tested will consist of a graphic showing the response time obtained as the number of clients is increased, remarking the point (number of simultaneous clients) where the performance requirement, if any, is not met or the error rate (which will also have to be specified) goes beyond a prefixed/acceptable percentage for every test case.

Although the scope of this document's test cases is related to server testing (excluding the client layer), these requirements will be considered in the test case definition procedure because they will provide with interesting inputs for the later acceptance test cases definition phase.

Almost all scalability tests can be run in a similar way as the performance tests, with the ANT target *testScalability* added to the SVN repository for CSA. Before running the tests, some configuration parameters must be tuned. These parameters are commented out in the file *build.properties.defaults* for the UI application in the client layer. The parameters in the general section are the same ones that have been explained in the 7.18 chapter. The parameters that only apply for the scalability tests are shown below:

- *conf.scalability.dir=test/scalability*. This parameter sets the directory where the script templates, etc. are. It should be left like it is.
- *build.scalability.dir=\${build.dir}/scalability*. This parameter sets the directory where the scalability tests are going to be run. It should be left like it is.
- *Scalability.tests="NameOfClass1:testMethod1 NameOfClass2:testMethod2"*. This parameter sets the tests that are going to be run in a similar way it was done for the performance tests.
- *scalability.num.workers=1*. This is the number of worker processes that will be started in every agent. It may be increased one by one to actually get numbers about the scalability of the system for every test executed.



- *scalability.agents="satlo7.net4.lan satlo5.net4.lan"*. This parameter sets up the agents that are going to be started for performing the test. It will be increased one by one to perform the scalability tests. In the end, the number of clients will be the number of agents multiplied by the number of workers that will be started in every agent.

7.19.1 Test case UI-SCALABILITY-06: Scalability of direct download retrieval close to maximum size allowed

This test performs a scalability analysis for a typical workload scenario for retrievals of data close to the maximum size allowed for direct download (1Gb).

7.19.1.1 Inputs specification

Select dataset(s) to direct download for 10 days.

7.19.1.2 Outputs specification

The result is visible in the Grinder console just after the execution of the test.

7.19.1.3 Test pass-fail criteria

Not applicable.

7.19.1.4 Environmental needs

Not applicable.

7.19.1.5 Special procedural constraints

This test requires that the user is logged in.

7.19.1.6 Interface dependencies

None.

7.19.2 Test case UI-SCALABILITY-07: Scalability of Shopping Basket retrieval close to maximum size allowed

This test performs a scalability analysis for a typical workload scenario for retrievals of data close to the maximum size allowed for Shopping Basket (40Gb).

7.19.2.1 Inputs specification

Select dataset(s) to download via Shopping Basket for 10 days.

7.19.2.2 Outputs specification

The result is visible in the Grinder console just after the execution of the test.

**7.19.2.3 Test pass-fail criteria**

Not applicable.

7.19.2.4 Environmental needs

Not applicable.

7.19.2.5 Special procedural constraints

This test requires that the user is logged in.

7.19.2.6 Interface dependencies

None.

7.19.3 Test case UI-SCALABILITY-09: Scalability of typical workload scenario for a short time range

This test performs a scalability analysis for a typical workload scenario for queries and retrievals with short time range as criteria.

7.19.3.1 Inputs specification

Enter as time range 2 minutes.

7.19.3.2 Outputs specification

The result is visible in the Grinder console just after the execution of the test.

7.19.3.3 Test pass-fail criteria

Not applicable.

7.19.3.4 Environmental needs

Not applicable.

7.19.3.5 Special procedural constraints

This test requires that the user is logged in.

7.19.3.6 Interface dependencies

None.

7.19.4 Test case UI-SCALABILITY-10: Scalability of typical workload scenario for a large time range

This test performs a scalability analysis for a typical workload scenario for queries and retrievals with large time range as criteria.

7.19.4.1 Inputs specification

Enter as time range 2 months.

7.19.4.2 Outputs specification

The result is visible in the Grinder console just after the execution of the test.

7.19.4.3 Test pass-fail criteria

Not applicable.

7.19.4.4 Environmental needs

Not applicable.

7.19.4.5 Special procedural constraints

This test requires that the user is logged in.

7.19.4.6 Interface dependencies

None.

7.20 UI-STRESS Test Cases***7.20.1 Test case UI-STRESS-01: Stress Server Layer with a typical workload scenario***

This test performs an analysis for a situation of a high workload on the system. The test aims at identifying the behaviour of the system when the load is at the maximum level the system may acceptably support. This level has been found out in the scalability tests.

7.20.1.1 Inputs specification

Maximum load level the system may acceptably support should be determined from the scalability tests first.

7.20.1.2 Outputs specification

The result is visible in the Grinder console just after the execution of the test and also by using monitoring tools in the machines where the system is running on.

7.20.1.3 Test pass-fail criteria

Test passed if application shows the ability to function acceptably during a high workload scenario.

7.20.1.4 Environmental needs

Not applicable.

7.20.1.5 Special procedural constraints

This test requires that the user is logged in.

7.20.1.6 Interface dependencies

None.

7.20.2 Test case *UI-STRESS-02: Recoverability of the system*

This test performs an analysis of the recoverability of the system for a situation where the system crashes after a very high workload. The test aims at identifying the problems encountered when bringing back the system to operations and also the causes of the crash (memory, CPU etc).

7.20.2.1 Inputs specification

Maximum load level the system may acceptably support should be determined from the scalability tests first.

7.20.2.2 Outputs specification

The result is visible in the Grinder console just after the execution of the test and also by using monitoring tools in the machines where the system is running on.

7.20.2.3 Test pass-fail criteria

Test passed if application recovers in a manner deemed acceptable.

7.20.2.4 Environmental needs

Not applicable.

7.20.2.5 Special procedural constraints

This test requires that the user is logged in.

7.20.2.6 Interface dependencies

None.

7.21 CAIO-BASIC Tests Cases

7.21.1 Test Case *CAIO-BASIC-01: Web pages load and display in common browsers on Linux OS*

This test validates opening the CAIO web pages in each of the most common browsers in a Linux OS environment. It checks for compatibilities problems with the browser and that the all the pages can be viewed correctly.



7.21.1.1 Inputs specification

Use different browsers to access to CAIO web pages.

7.21.1.2 Outputs specification

The CAIO web pages are loaded and displayed correctly and the CSS is adhered to.

7.21.1.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.21.1.4 Environmental needs

This test case shall be executed in a Linux OS environment with common browsers.

7.21.1.5 Special procedural constraints

Not applicable.

7.21.1.6 Interface dependencies

None.

7.21.2 Test Case CAIO-BASIC-02: Web pages load and display in common browsers on Windows OS

This test validates opening the CAIO web pages in each of the most common browsers in a Windows OS environment. It checks for compatibilities problems with the browser and that the all the pages can be viewed correctly.

7.21.2.1 Inputs specification

Use different browsers to access to CAIO web pages.

7.21.2.2 Outputs specification

The CAIO web pages are loaded and displayed correctly and the CSS is adhered to.

7.21.2.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.21.2.4 Environmental needs

This test case shall be executed in a Windows OS environment with common browsers.

7.21.2.5 Special procedural constraints

Not applicable.

7.21.2.6 Interface dependencies

None.

7.21.3 Test Case CAIO-BASIC-03: Web pages load and display in common browsers on Mac OS

This test validates opening the CAIO web pages in each of the most common browsers in a Mac OS environment. It checks for compatibilities problems with the browser and that the all the pages can be viewed correctly.

7.21.3.1 Inputs specification

Use different browsers to access to CAIO web pages.

7.21.3.2 Outputs specification

The CAIO web pages are loaded and displayed correctly and the CSS is adhered to.

7.21.3.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.21.3.4 Environmental needs

This test case shall be executed in a Mac OS environment with common browsers.

7.21.3.5 Special procedural constraints

Not applicable.

7.21.3.6 Interface dependencies

None.

7.21.4 Test Case CAIO-BASIC-04: Web pages load and display outside the ESAC firewall

This test validates opening the CAIO web pages in an environment outside the ESAC firewall and under network restrictive environments where only standard HTTP port 80 is opened.

7.21.4.1 Inputs specification

Use different browsers to access to CAIO web pages.

7.21.4.2 Outputs specification

The CAIO web pages are loaded and displayed correctly and the CSS is adhered to.

7.21.4.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.



7.21.4.4 Environmental needs

Client machine must be outside ESAC firewall and with outgoing port 80 opened only (note: use for this 'sattest01.esac.esa.int', user 'satops').

7.21.4.5 Special procedural constraints

Not applicable.

7.21.4.6 Interface dependencies

None.

7.21.5 Test Case CAIO-BASIC-05: CAIO menu bar functionality

This test validates that the navigation menu in the CAIO web pages functions correctly and takes the user to the correct pages.

7.21.5.1 Inputs specification

Select all the navigation menu items one after the other.

7.21.5.2 Outputs specification

The page corresponding to the menu item selected is loaded correctly.

7.21.5.3 Test pass-fail criteria

Test passed if the correct pages load when selected from the navigation menu, failed otherwise.

7.21.5.4 Environmental needs

None.

7.21.5.5 Special procedural constraints

Not applicable.

7.21.5.6 Interface dependencies

None.

7.21.6 Test Case CAIO-BASIC-06: CAIO links

This test validates that every link provided on CAIO webpages points to the correct web page.

7.21.6.1 Inputs specification

Follow all the links embedded in the CAIO web pages.



7.21.6.2 Outputs specification

The page corresponding to the link selected is loaded correctly.

7.21.6.3 Test pass-fail criteria

Test passed if the links are not broken, failed otherwise.

7.21.6.4 Environmental needs

None.

7.21.6.5 Special procedural constraints

Not applicable.

7.21.6.6 Interface dependencies

None.

7.21.7 Test Case CAIO-BASIC-07: Login requested

This test validates that, when the user is being prompted to log in, the login popup is displayed correctly. If the popup is blocked, then this test verifies that a suitable notification is displayed alerting the user to this fact.

7.21.7.1 Inputs specification

Make a HTTP request (product request) to the CAIO from each of the most common browsers.

7.21.7.2 Outputs specification

A logging pop-up window is displayed correctly or a suitable popup-blocked notification is displayed.

7.21.7.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.21.7.4 Environmental needs

None.

7.21.7.5 Special procedural constraints

User not logged in, and not cookies with user credentials are stored at the browser.

7.21.7.6 Interface dependencies

None.



7.21.8 Test Case CAIO-BASIC-o8: AIO client on Linux OS

This test validates that the AIO command line client functions correctly in a Linux OS environment.

7.21.8.1 Inputs specification

Download the AIO client from CAIO onto a Linux OS and deploy it. Execute it with some of the examples provided at the CAIO web pages.

7.21.8.2 Outputs specification

Data is retrieved as explained at the CAIO web pages.

7.21.8.3 Test pass-fail criteria

Test passed if AIO command line client works correctly, failed otherwise.

7.21.8.4 Environmental needs

Java installation must be available.

7.21.8.5 Special procedural constraints

User must have an account on RSSD LDAP system.

7.21.8.6 Interface dependencies

None.

7.21.9 Test Case CAIO-BASIC-o9: AIO client on Windows OS

This test validates that the AIO command line client functions correctly in a Windows OS environment.

7.21.9.1 Inputs specification

Download the AIO client from CAIO onto a Windows OS and deploy it. Execute it with some of the examples provided at the CAIO web pages.

7.21.9.2 Outputs specification

Data is retrieved as explained at the CAIO web pages.

7.21.9.3 Test pass-fail criteria

Test passed if AIO command line client works correctly, failed otherwise.

7.21.9.4 Environmental needs

Java installation must be available.

7.21.9.5 Special procedural constraints

User must have an account on RSSD LDAP system.

7.21.9.6 Interface dependencies

None.

7.21.10 Test Case CAIO-BASIC-10: AIO client on Mac OS

This test validates that the AIO command line client functions correctly in a Mac OS environment.

7.21.10.1 Inputs specification

Download the AIO client from CAIO onto a Mac OS and deploy it. Execute it with some of the examples provided at the CAIO web pages.

7.21.10.2 Outputs specification

Data is retrieved as explained at the CAIO web pages.

7.21.10.3 Test pass-fail criteria

Test passed if AIO command line client works correctly, failed otherwise.

7.21.10.4 Environmental needs

Java installation must be available.

7.21.10.5 Special procedural constraints

User must have an account on RSSD LDAP system.

7.21.10.6 Interface dependencies

None.

7.21.11 Test Case CAIO-BASIC-11: Authentication via ‘wget’

This test validates that a user can be authenticated using ‘wget’ and that the credentials can be stored in a cookie for further use.

7.21.11.1 Inputs specification

Use the login request with ‘wget’ as explained at CAIO web pages.

7.21.11.2 Outputs specification

A file corresponding to a cookie is created at the local directory and a ‘credentials.log’ file is downloaded and stored at the same directory containing status of the authentication



(successful or failed). If authentication was successful, the cookie file contains user data, if not, the cookie file is empty (only comments).

7.21.11.3 Test pass-fail criteria

Test passed if the cookie file and credentials file are successfully stored at local directory, failed otherwise.

7.21.11.4 Environmental needs

Wget command must be available.

7.21.11.5 Special procedural constraints

User must have an account on RSSD LDAP system.

7.21.11.6 Interface dependencies

None.

7.22 CAIO-SEARCH Test Cases

7.22.1 Test case CAIO-SEARCH-01: No data returned notification

This test validates that, when performing a product query that it is known not to exist, the appropriate notification is displayed to the user informing that no data was found.

7.22.1.1 Inputs specification

1- Perform a product query with criteria for which there are not results.

Ex: `http://cfa.esac.esa.int/cfa/aio/product-action?DATASET_ID=*WBD*&START_DATE=2004-10-01T11:35:00Z&END_DATE=2004-10-01T18:35:00Z`

2- Perform a product query with criteria for which there are not results, with NON_BROWSER option.

3- Perform a product query with criteria for which there are not results via 'wget'.

7.22.1.2 Outputs specification

1- Browser is redirected to a page displaying error message of HTTP status 404 "No results found".

2- Browser is redirected to a page displaying error message of HTTP status 404 "No results found".

3- Response returned: "ERROR 404: Not found".

7.22.1.3 Test pass-fail criteria

Test passed if a notification is displayed stating that no data were found, failed otherwise.

7.22.1.4 Environmental needs

Not applicable.

7.22.1.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.22.2 Test case CAIO-SEARCH-02: No metadata returned notification

This test validates that, when performing a metadata query known to not contain any results, the appropriate notification is displayed to the user informing that no metadata was found.

7.22.2.1 Inputs specification

Perform a metadata query with criteria for which there are not results from browser, with option NON_BROWSER and via 'wget'.

7.22.2.2 Outputs specification

In the three cases, depending on the RETURN_TYPE specified, if VOTable, then returns an empty table with 0 results; if JSON, then the response contains 0 as number of results and an empty data structure; if CSV an empty csv file is returned.

7.22.2.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.22.2.4 Environmental needs

Not applicable.

7.22.2.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.22.3 Test case CAIO-SEARCH-03: Metadata queries

This test validates that the correct number and type of results are returned for every combination of metadata query.

7.22.3.1 Inputs specification

Perform a metadata query with all possible combinations of parameters from browser and via 'wget'.

7.22.3.2 Outputs specification

The correct metadata results are returned for every query specified.

7.22.3.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.22.3.4 Environmental needs

Not applicable.

7.22.3.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.22.4 Test case CAIO-SEARCH-04: Product queries

This test validates that the correct data file(s) is(are) returned for every combination of product data query.

7.22.4.1 Inputs specification

- 1- Perform a product request with all possible combinations of parameters in synchronous mode.
- 2- Perform a product request with all possible combinations of parameters in asynchronous mode.

7.22.4.2 Outputs specification

The correct data file(s) is(are) returned for every request specified.

7.22.4.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.22.4.4 Environmental needs

Not applicable.

7.22.4.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.22.5 Test case CAIO-SEARCH-05: Metadata queries Return Type

This test validates that the results of a metadata query are returned in the correct format when the RETURN_TYPE is specified. It also checks that, if RETURN_TYPE is not specified, the default format is VOTable.

7.22.5.1 Inputs specification

Perform a metadata query specified RETURN_TYPE equals to “JSON”; equals to “CSV”; equals to “VOTable” and without RETURN_TYPE specified.

7.22.5.2 Outputs specification

The metadata results are returned in the correct file format.

7.22.5.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.22.5.4 Environmental needs

Not applicable.

7.22.5.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.22.6 Test case CAIO-SEARCH-06: Metadata queries constrained by PAGE_SIZE and PAGE

This test validates that the returned metadata results have the correct number of results when the PAGE_SIZE and PAGE parameters are specified in the query.

7.22.6.1 Inputs specification

Perform a metadata query specifying PAGE_SIZE and PAGE.

7.22.6.2 Outputs specification

The metadata results returned are a subset of the metadata query results corresponding to the page indicated and as many items as PAGE_SIZE.

7.22.6.3 Test pass-fail criteria

Test passed if the output is as expected, failed otherwise.

7.22.6.4 Environmental needs

Not applicable.

7.22.6.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.



7.22.7 Test case CAIO-SEARCH-07: Complex metadata queries with QUERY parameter

This test validates that the correct metadata results are returned when a query is built using the QUERY= parameter.

7.22.7.1 Inputs specification

Perform the metadata queries with QUERY parameter that appear as example at the CAIO Metadata web page.

7.22.7.2 Outputs specification

The metadata results returned match the expectations declared at the CAIO web page.

7.22.7.3 Test pass-fail criteria

Test passed if the correct number of metadata results is returned, failed otherwise.

7.22.7.4 Environmental needs

Not applicable.

7.22.7.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.22.8 Test case CAIO-SEARCH-08: Sorting metadata queries

This test validates that the returned metadata results are in the correct order when the ORDER_BY parameter is specified.

7.22.8.1 Inputs specification

Take the metadata query at the CAIO web page that queries for files and add &ORDER_BY=FILE.START_DATE.

Example:

```
http://cfa.esac.esa.int/cfa/aio/metadata-
action?SELECTED_FIELDS=FILE.FILE_NAME,FILE.START_DATE,FILE.END_DATE,FILE.CAA_ING
ESTION_DATE&RESOURCE_CLASS=FILE&QUERY=(DATASET.DATASET_ID like 'C1_CP_EDI_EGD'
OR DATASET.DATASET_ID like 'C1_CP_EFW_L3_P' OR DATASET.DATASET_ID like
'C1_CP_FGM_FULL') AND FILE.START_DATE <= '2001-02-05' AND FILE.END_DATE >=
'2001-02-01'&PAGE_SIZE=1000&PAGE=1&ORDER_BY=FILE.START_DATE&RETURN_TYPE=JSON
```

7.22.8.2 Outputs specification

The metadata results returned are sorted by start date of the file instead of by file name.

7.22.8.3 Test pass-fail criteria

Test passed if the returned metadata results are in the correct order, failed otherwise

7.22.8.4 Environmental needs

Not applicable.

7.22.8.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.22.9 Test case CAIO-SEARCH-09: Header Requests

This test validates that for data requests, specifying RETRIEVALTYPE=HEADER results in the returned file being the XML header file (s) of the relevant dataset.

7.22.9.1 Inputs specification

Perform a header request for certain datasets.

7.22.9.2 Outputs specification

A XML file per dataset selected containing the dataset metadata.

7.22.9.3 Test pass-fail criteria

Test passed if the relevant XML file(s) is returned, failed otherwise

7.22.9.4 Environmental needs

Not applicable.

7.22.9.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.22.10 Test case CAIO-SEARCH-10: Wrong query format

This test validates that if a specified query is badly formed or defined then the relevant notification is displayed to the user.

7.22.10.1 Inputs specification

- 1- Perform a data request with a wrong parameter name.
- 2- Perform a data request with a wrong date format.
- 3- Perform a data request with wrong values for some parameters like DELIVERY_INTERVAL.
- 4- Repeat previous steps via 'wget'.

7.22.10.2 Outputs specification

- 1- If “DATASET_ID” is badly spelt, an error page with message that mandatory field is missing is displayed. If any other parameter is misspelt, the parameter value is ignored.
- 2- If a date value is badly entered, an error page with message that the date cannot be parsed is displayed.
- 3- If ‘hour’ is specified instead of ‘Hourly’ for Delivery Interval, an error page with message that the value for that parameter is invalid.
- 4- Output for all steps are the same: “ERROR 400: Bad request”.

7.22.10.3 Test pass-fail criteria

Test passed if a notification is displayed stating that the query was badly formed or defined, failed otherwise.

7.22.10.4 Environmental needs

Not applicable.

7.22.10.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.23 CAIO-RESULTS Test Cases

7.23.1 Test case CAIO-RESULTS-01: Results in VOTable

This test validates that the file returned from a metadata query (for ex. Inventory request) that specifies a RETURN_TYPE of VOTable is in the VOTable file format. It also tests that results are correctly formatted in the file and that the pagination and number of records have been respected.

7.23.1.1 Inputs specification

Perform an inventory request (for example the second request at the CAIO Inventory page).

7.23.1.2 Outputs specification

A file in VOTable format is returned. The results are correctly formatted in the file and the pagination and number of records have been respected (for the example, 16 records).

7.23.1.3 Test pass-fail criteria

Test passed if the returned metadata file is in the correct file format and the contents formatted correctly, failed otherwise.



7.23.1.4 Environmental needs

Not applicable.

7.23.1.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.23.2 Test case CAIO-RESULTS-02: Results in JSON

This test validates that the file returned from a metadata query (for ex. Inventory request) that specifies a RETURN_TYPE of JSON is in the JSON file format. It also tests that results are correctly formatted in the file and that the pagination and number of records have been respected.

7.23.2.1 Inputs specification

Perform an inventory request (for example first request at the CAIO Inventory page).

7.23.2.2 Outputs specification

A file in JSON format is returned. The results are correctly formatted in the file and the pagination and number of records have been respected (in the example 28 records).

7.23.2.3 Test pass-fail criteria

Test passed if the returned metadata file is in the correct file format and the contents formatted correctly, failed otherwise.

7.23.2.4 Environmental needs

Not applicable.

7.23.2.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.23.3 Test case CAIO-RESULTS-03: Results in CSV

This test validates that the file returned from a metadata query (for ex. Inventory request) that specifies a RETURN_TYPE of CSV is in the CSV file format. It also tests that results are correctly formatted in the file and that the pagination and number of records have been respected.

7.23.3.1 Inputs specification

Perform an inventory request (for example one of the requests at the CAIO Inventory page).



7.23.3.2 Outputs specification

A file in CSV format is returned. The results are correctly formatted in the file and the pagination and number of records have been respected.

7.23.3.3 Test pass-fail criteria

Test passed if the returned metadata file is in the correct file format and the contents formatted correctly, failed otherwise.

7.23.3.4 Environmental needs

Not applicable.

7.23.3.5 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.24 CAIO-RETRIEVAL Test Cases

7.24.1 Test case CAIO-RETRIEVAL-01: Synchronous retrieve of data product

This test validates that a file can be correctly retrieved as the result of a product data request (synchronous).

7.24.1.1 Inputs specification

Perform a data product request.

7.24.1.2 Outputs specification

A download requested window appears. Once data is ready, a save dialog box is displayed and the file(s) can be saved into disk.

7.24.1.3 Test pass-fail criteria

Test passed if file(s) is successfully retrieved and saved locally to disk, failed otherwise.

7.24.1.4 Environmental needs

Software able to uncompress and untar tar.gz files is required (Example: “gunzip” and “tar xvf ...”).

7.24.1.5 Special procedural constraints

User must log in or having stored a session cookie with its user credentials.

7.24.1.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.24.2 Test case CAIO-RETRIEVAL-02: Retrieve of header files

This test validates that a XML dataset metadata file can be correctly retrieved as the result of a header request which is a data request with REQUESTTYPE=HEADER.

7.24.2.1 Inputs specification

Perform a header request.

7.24.2.2 Outputs specification

Once data is ready, a save dialog box is displayed and the file can be saved into disk. The file is a zip file that contains a XML file per dataset selected.

7.24.2.3 Test pass-fail criteria

Test passed if file is successfully retrieved and saved locally to disk, failed otherwise.

7.24.2.4 Environmental needs

Software able to uncompress zip files is required (Example: “unzip”).

7.24.2.5 Special procedural constraints

User must log in or having stored a session cookie with its user credentials.

7.24.2.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.24.3 Test case CAIO-RETRIEVAL-03: Retrieve of data products from multiple datasets

This test validates that a single compressed file containing the files from a group of datasets can be correctly retrieved as the result of a product data request (synchronous).

7.24.3.1 Inputs specification

Perform a data product request that include several datasets (note that wild cards are allowed at DATASET_ID parameter).

7.24.3.2 Outputs specification

A download requested window appears. Once data is ready, a save dialog box is displayed and the file can be saved into disk.

7.24.3.3 Test pass-fail criteria

Test passed if compressed file of a group of files is successfully retrieved and saved locally to disk, failed otherwise.

7.24.3.4 Environmental needs

Software able to uncompress and untar tar.gz files is required (Example: “gunzip” and “tar xvf ...”).

7.24.3.5 Special procedural constraints

User must log in or having stored a session cookie with its user credentials.

7.24.3.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.24.4 Test case CAIO-RETRIEVAL-04: Retrieve of Inventory data

This test validates that inventory information about a dataset can be correctly retrieved as the result of a metadata query that has specified DATASET_INVENTORY as SELECT_FIELDS and RESOURCE_CLASS parameters.

7.24.4.1 Inputs specification

Perform an inventory request (not specifying RETURN_TYPE).

7.24.4.2 Outputs specification

Once data is ready, a save dialog box is displayed and a file in VOTable format can be saved into disk.

7.24.4.3 Test pass-fail criteria

Test passed if the correct inventory information is successfully retrieved and saved locally to disk, failed otherwise.

7.24.4.4 Environmental needs

None.

7.24.4.5 Special procedural constraints

Not applicable.

7.24.4.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.24.5 Test case CAIO-RETRIEVAL-05: Asynchronous retrieval of data product

This test validates that a file can be correctly retrieved as the result of an asynchronous product data request.

7.24.5.1 Inputs specification

Perform an asynchronous data product request.

7.24.5.2 Outputs specification

A web page is displayed providing information about where the data will be located once ready and if an e-mail will be sent to the user or not.

In the e-mail received by the user the link to the location is included.

7.24.5.3 Test pass-fail criteria

Test passed if the data can be retrieved successfully from the location and saved locally to disk, failed otherwise.

7.24.5.4 Environmental needs

Software able to uncompress and untar tar.gz files is required (Example: “gunzip” and “tar xvf ...”).

7.24.5.5 Special procedural constraints

User must log in or having stored a session cookie with its user credentials.

7.24.5.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.24.6 Test case CAIO-RETRIEVAL-06: Asynchronous retrieval of data product from multiple datasets

This test validates that a single compressed file containing the files from a group of datasets can be correctly retrieved as the result of an asynchronous product data request.

7.24.6.1 Inputs specification

Perform an asynchronous data product request that include several datasets.

7.24.6.2 Outputs specification

A web page is displayed providing information about where the data will be located once ready and if an e-mail will be sent to the user or not.

In the e-mail receive by the user the link to the location is included.



7.24.6.3 Test pass-fail criteria

Test passed if a compressed file can be retrieved successfully from the location and saved locally to disk, failed otherwise.

7.24.6.4 Environmental needs

Software able to uncompress and untar tar.gz files is required (Example: “gunzip” and “tar xvf ...”).

7.24.6.5 Special procedural constraints

User must log in or having stored a session cookie with its user credentials.

7.24.6.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.24.7 Test case CAIO-RETRIEVAL-07: CAIO user manual

This test validates that the User Manual can be successfully downloaded from the CAIO web pages and that it is the most recent version.

7.24.7.1 Inputs specification

Follow the link at the CAIO Manual page.

7.24.7.2 Outputs specification

A dialog box either to open the manual document or to save it in disk is displayed.

7.24.7.3 Test pass-fail criteria

Test passed if the User Manual is successfully retrieved and saved locally to disk, failed otherwise.

7.24.7.4 Environmental needs

Adobe installed in order to open the manual.

7.24.7.5 Special procedural constraints

Not applicable.

7.24.7.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.



7.24.8 Test case CAIO-RETRIEVAL-08: AIO command line client

This test validates that the AIO command line client can be successfully downloaded from the CAIO web pages and that it is the most recent version.

7.24.8.1 Inputs specification

Follow the link at the CAIO Java Client page.

7.24.8.2 Outputs specification

Save the tar in local disk and deploy it. AIO command line client ready to be used.

7.24.8.3 Test pass-fail criteria

Test passed if the command line client is successfully retrieved and saved locally to disk, failed otherwise.

7.24.8.4 Environmental needs

Java runtime environment must be installed.

7.24.8.5 Special procedural constraints

Not applicable.

7.24.8.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.24.9 Test case CAIO-RETRIEVAL-09: Retrieve metadata files

This test validates that metadata can be successfully retrieved in the correct file format when the RETURN_TYPE is specified for: VOTable, JSON and CSV.

7.24.9.1 Inputs specification

Perform metadata requests and choose to save the results in a file.

7.24.9.2 Outputs specification

A file of the correct format with the metadata request results is stored at local disk.

7.24.9.3 Test pass-fail criteria

Test passed if metadata is successfully retrieved and saved locally to disk in the correct file format, failed otherwise.

7.24.9.4 Environmental needs

Java runtime environment must be installed.

7.24.9.5 Special procedural constraints

Not applicable.

7.24.9.6 Interface dependencies

Either test case CAIO-BASICS-01, CAIO-BASICS-02, CAIO-BASICS-03, CAIO -BASICS-04 or CAIO -BASICS-05 must be executed before.

7.25 CAIO-LOGGING Test Cases

7.25.1 Test case CAIO-LOGGING-01: Log output

This test validates whether the logs are correctly written to the log file in the location of the webapps files for the relevant deployment environment.

7.25.1.1 Inputs specification

Go to the location of the webapps files and search for the log files.

7.25.1.2 Outputs specification

A log file for the AIO subsystem exists.

7.25.1.3 Test pass-fail criteria

Test passed if the logs are correctly written in the correct location, failed otherwise.

7.25.1.4 Environmental needs

Not applicable.

7.25.1.5 Interface dependencies

None.

7.25.2 Test case CAIO-LOGGING-02: Logging levels

This test validates whether changing the logging level correctly changes the detail of logs provided.

7.25.2.1 Inputs specification

Change the log level at the CAIO from INFO to DEBUG.

7.25.2.2 Outputs specification

More detailed logging is performed.



7.25.2.3 Test pass-fail criteria

Test passed if detail of logging output correctly changes when different levels are specified, failed otherwise.

7.25.2.4 Environmental needs

Not applicable.

7.25.2.5 Interface dependencies

None.

7.26 CAIO-SECURITY Test Cases

7.26.1 Test case CAIO-SECURITY-01: Log in/ logout

This test validates that the user can be authenticated against the RSSD LDAP system and that the login/logout functionality work successfully.

7.26.1.1 Inputs specification

- 1- Log in the CAIO using RSSD LDAP credentials (username and password) using login request as explained in the manual.
- 2- Logout from CAIO using logout request as explained in the manual or deleting the cookies at the browser or closing the browser.

7.26.1.2 Outputs specification

- 1- A message informing about the user successful login is displayed at a new browser window.
- 2- If logout request used a message informs about the current user who is anonymous.

7.26.1.3 Test pass-fail criteria

Test passed if the user is authenticated against the RSSD LDAP system correctly, failed otherwise.

7.26.1.4 Environmental needs

Not applicable.

7.26.1.5 Special procedural constraints

User need to be registered at the RSSD LDAP registry.

7.26.1.6 Interface dependencies

None.



7.26.2 Test case CAIO-SECURITY-02: Access to public services

This test validates that a user who has not logged in only has access to perform inventory requests and metadata requests.

7.26.2.1 Inputs specification

- 1- Perform inventory and metadata requests.
- 2- Try to perform a data request or an asynchronous data request.

7.26.2.2 Outputs specification

- 1- Inventory and metadata requests are performed successfully.
- 2- The login popup window is displayed or a message is returned, requesting to enter RSSD LDAP credentials.

7.26.2.3 Test pass-fail criteria

Test passed if logged out user only has accessed to public data, failed otherwise.

7.26.2.4 Environmental needs

Not applicable.

7.26.2.5 Special procedural constraints

This test requires that the user is not logged in.

7.26.2.6 Interface dependencies

None.

7.26.3 Test case CAIO-SECURITY-03: Access to non-public services

This test validates that a non-logged user is requested to login when attempting to perform any non-public task. These tasks include: data product requests, data header requests, asynchronous product requests.

7.26.3.1 Inputs specification

- 1- At CAIO “Data” page, click on any product request.
- 2- At CAIO “Async Data” page, click on any asynchronous product request.

7.26.3.2 Outputs specification

In both cases a popup window appears or a message is returned requesting to log in.

7.26.3.3 Test pass-fail criteria

Test passed if the user is requested to log in when attempting to perform any non-public task, failed otherwise.



7.26.3.4 Environmental needs

Not applicable.

7.26.3.5 Special procedural constraints

This test requires that the user is not logged in.

7.26.3.6 Interface dependencies

None.

8 SOFTWARE VALIDATION TEST PROCEDURES

The software validation test procedures are out of the scope of this document. If needed, they will be described in a separate document.

9 SOFTWARE VALIDATION ANALYSIS, INSPECTION, REVIEW OF DESIGN

The test cases listed below cannot be fully validated through test. Thus in addition they are validated through inspection of their output log files:

- Test Case IMPORT-BASIC-01: Start-up of Import server
- Test Case IMPORT-BASIC-02: Import directories available
- Test Case IMPORT-BASIC-03: Ingestion Request File validity
- Test Case IMPORT-BASIC-04: Sorting of Ingestion Request Files
- Test Case IMPORT-BASIC-05: Ingestion Request File format
- Test Case IMPORT-BASIC-06: Import Data files
- Test Case IMPORT-BASIC-07: Import Header files
- Test Case IMPORT-BASIC-08: Import Metadata files
- Test Case IMPORT-BASIC-09: Ingest queue
- Test Case INGEST-BASIC-01: Start-up of Ingest server
- Test Case INGEST-BASIC-02: Daemon process
- Test Case INGEST-BASIC-03: Associated import logs
- Test Case INGEST-BASIC-04: Ingest Data files
- Test Case INGEST-BASIC-05: Ingest XML metadata files
- Test Case INGEST-BASIC-06: Ingest Header files
- Test Case INGEST-BASIC-07: Ingest Inventory files
- Test Case INGEST-BASIC-08: Ingest Graphical Metadata CSV files
- Test Case DD-BASIC-01: Start-up of Data Distribution service
- Test Case DD-BASIC-02: Port open for Data Distribution service
- Test Case GRAPHIC-BASIC-01: Generate On-demand plot
- Test Case GRAPHIC-BASIC-02: Generate pre-generated plot

10 VALIDATION TEST PLATFORMS REQUIREMENTS

10.1 Performance, scalability and stress testing considerations

It is important to remark that the main goal of testing is to find faults in the software system (and therefore they must be designed to that extent). However, the tests themselves may also give inputs for the assessment of performance, scalability, reliability, etc. which may be the case for performance, scalability and stress testing.

As a general background from the end user point of view, some usability research shows that for web applications (and in general any GUI application), the response time should always be less than or equal to 7 seconds. Even if the Cluster Science Archive has its own requirements for this matter, this and the following usability concerns shall be taken into account for designing the test cases:

- Simple operations should have a response time of 2 seconds or less.
- Per cent-done indicators are required when long running interactions because site abandonment tends to occur between 11 and 16 seconds.
- For incremental display transactions, useful content needs to load within 2 seconds and the operations must complete within 20-30 seconds.

The time (“think time”) that a user takes to start one operation (due to the fact that a form must be filled in or any other concern related to user interaction) must be considered when designing scalability and stress test cases, and also when modelling the overall performance of the system, as short “think times” will increase the load on the system and long “think times” will decrease it. This feature is then crucial for setting the time interval between two occurrences of the same test instance. For complex tests where there is more than one interaction with the system, this “think time” must be considered and should be taken into account in the test case implementation if accurate measurements have to be compiled.

The Requirements Baseline, in its User Requirements Document [CFA_EST_UR], specifies a set of performance requirements that shall be used to validate test results for performance, scalability and stress tests.

10.2 The GRINDER

This appendix is only intended to show a general overview about the functioning of the Grinder java load testing framework. For more information, please go to its web page⁵.

⁵ <http://grinder.sourceforge.net/>

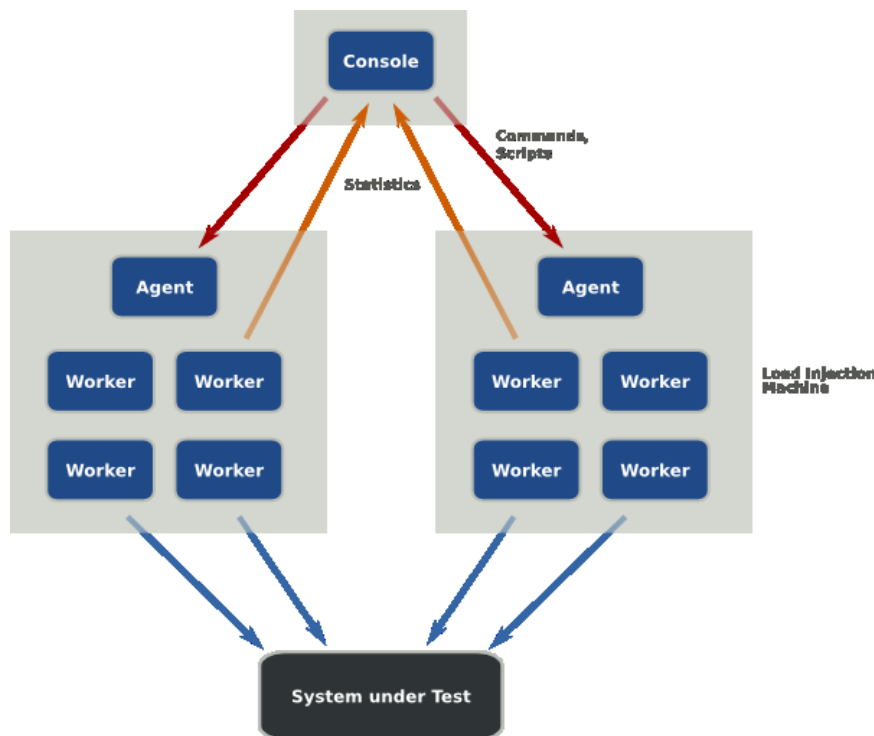


Figure 4: Grinder general architecture

Figure 4 shows the general architecture of Grinder, where there are several agents connected to the Grinder console (graphical user interface) which send the statistics of the tests executed by them and receive any command sent from the console. The system under test is of course outside of the Grinder architecture and normally resides in other machine(s) and/or network(s). Each agent can start several workers and each worker can start a number of threads. Each thread will then run one test instance configured in the grinder properties file (hidden to the user on the tests defined here for commodity purposes).

11 SOFTWARE VALIDATION SPECIFICATION ADDITIONAL INFORMATION

11.1 Requirement to test/analysis/inspection/review of design traceability matrix

P	N	REQ	Text	Test method	Test (T)	Analysis (A)	Inspection (I)	Review of Design (D)
General Requirements								
1	1	2.1.1	The CSA user interface should provide the same functionalities as CAA, offering access to a searchable Cluster data catalogue and displaying the requested variables, with the capability of further development based on user feedback and peer reviews.		X	X		
1	1	2.1.2	The CSA will provide services for on-demand sub-setting, format conversion and graphical product generation. The CSA should have sufficient computational and I/O capabilities to support these functions.		X	X		
1	1	2.1.3	The CSA will provide services for graphical product generation. The CSA should have sufficient computational and I/O capabilities to support these functions.		X	X		
1	1	2.1.4	The CSA should provide high speed access through the public internet		X	X		
1	1	2.1.5	The CSA should provide sufficient disk space to cope with the data storage requirements.				X	X
1	1	2.1.6	The users of the CSA must be registered and required to log in to the system before they can have access to the data holdings/plotting system.		X			
1	1	2.1.7	User can change the registration details, such as the user password, e-mail and address.		X			
1	1	2.1.8	User activity on CSA should be logged and monitored.		X	X		
2	1	2.1.9	CAA and CSA user repositories should be in synchronisation as long as CAA and CSA are run in parallel, including the user-id and password		X	X		
1	1	2.1.10	The CAA-CSA interface should allow the transfer of cluster raw data, cluster data files and datasets, cluster metadata, calibrated datasets of Double Star mission and other non-cluster datasets from CAA to CSA.		X	X		
Cluster Archive Datasets								
1	1	2.2.1	CSA should support data file and dataset version control (include the ability to ingest a new version of files).		X	X		
1	1	2.2.2	CSA should be able to add new datasets to the archive.		X			
1	1	2.2.3	CSA should be able to make datasets inactive or to remove them from the archive when necessary.		X		X	
1	1	2.2.4	To monitor the CAA data delivery and the related database ingestion activity on the CSA		X	X		
1	1	2.2.5	CSA should support the delivery of reference datasets (caveats, etc) to the users that are specified in the metadata of a requested dataset.		X			
Cluster Raw Data System								
3	1	2.3.1	CSA should provide to the users an access to the Cluster and Double Star raw data.		X			
Double Star and other non-cluster Mission Datasets								
3	1	2.4.1	CSA should provide to the users access to the calibrated datasets of the Double Star mission and other non-Cluster datasets.		X			
Documentation of the CFA								
3	3	2.5.1	CSA should provide online access to the documentation of the Cluster mission with a search capability.		X		X	
2	1	2.5.2	CSA should provide and maintain the user guide of the CSA system.		X		X	

P	N	REQ	Text	Test method	Test (T)	Analysis (A)	Inspection (I)	Review of Design (D)
Graphical User Interface								
1	1	3.1.1	CSA should provide a GUI access to the Cluster data.		X			
2	1	3.1.2	Context sensitive help should be available at the graphical user interface.		X		X	
1	1	3.1.3	During the dataset selection the user should easily be able to see the key parts of the dataset metadata		X		X	
1	1	3.1.4	CSA GUI should allow the user to query the cluster catalogue using the following search criteria: time interval (Start/Stop time), experiment, measurement type and instrument type.		X			
1	1	3.1.5	All search criteria may be combined in a search query.		X			
1	1	3.1.6	The datasets not available for the requested time interval should not be listed / made available to the user		X		X	
3	3	3.1.7	The data catalogue should be searchable on data values (e.g. value > x), spacecraft positions, spacecraft configuration and tetrahedron quality.		X			
1	1	3.1.8	The result of a query should be a list of the datasets satisfying the search criteria.		X	X		
1	1	3.1.9	It should be possible to save and load a query result.		X			
1	1	3.1.10	The parameters to display for each dataset of the query result should be: - Dataset title, - Observatory, - Dataset name (in a contextual menu), - Dataset time coverage (in a contextual menu).		X		X	
1	1	3.1.11	The datasets of a query result should be grouped by experiment at a first level.		X			
1	1	3.1.12	The datasets of a query result should be grouped in science and ancillary groups at a second level.		X			
2	1	3.1.13	It should be possible to expand/collapse the list of datasets of a query result at first and second group level.		X			
1	1	3.1.14	The datasets of a query result displayed within a science/ancillary group should be restricted according to units; only the datasets of one unit are shown at one time		X			
2	1	3.1.15	Key datasets should be displayed on top within their groups.		X		X	
2	1	3.1.16	It should be possible to filter the query result by string search at dataset title.		X			
1	1	3.1.17	For requesting data, it should be possible to select all datasets displayed at the query result page or to select them one by one per spacecraft (Cluster 1, Cluster 2, Cluster 3, Cluster 4 or multi-cluster).		X			
1	1	3.1.18	CSA users should be able to directly download the requested data from the query result page, if the amount of data is less than 1GB.		X			
1	1	3.1.19	CSA users should be able to select and deselect products they wish to retrieve using a "shopping basket".		X			
1	1	3.1.20	Profile: the user should be able to save and modify the selected datasets for requesting data, including the corresponding query request.		X			
Data Extraction and Preparation								
1	1	3.2.1	The user should be able to select the time granularity of the files for downloading between a number of options such as daily or hourly files, or a single file (per dataset).		X			
1	1	3.2.2	It must be possible to concatenate data from distinct files before delivery.		X			
1	1	3.2.3	The user should be able to select the delivery format of the calibrated data files for downloading, between CEF and CDF.		X			
2	2	3.2.4	CSA should display an estimation of the size of requested data.		X			
1	1	3.2.5	Processing of search requests must be completed within a reasonable time.		X	X		
1	1	3.2.6	CSA system should notify the user when the requested data is available for downloading.		X			
1	1	3.2.7	CSA system should keep the data available for downloading for 3 weeks.		X			

P	N	REQ	Text	Test method	Test (T)	Analysis (A)	Inspection (I)	Review of Design (D)
Data Delivery								
1	1	3.3.1	File compression & packaging should be applied for data delivery.		X			
1	1	3.3.2	Direct download should be limited to 1 GB. Command-line requests should be limited to 40GB.		X			
2	3	3.3.3	It should be possible to cancel a delivery request at any time during a direct download and before delivery has commenced when shopping-basket option was used.		X	X		
Command-line Interface								
1	1	3.4.1	To provide access via machine interface to the archive holdings in asynchronous ("batch") mode		X			
2	3	3.4.2	To provide access via machine interface to the archive holdings in synchronous mode (limited capability)		X			
3	3	3.4.3	To provide "data-streaming" service access to data catalogues and data retrieval capabilities		X			
Visualization Tools								
1	1	3.5.1	CSA should display the most popular graphical variables for the selected time intervals (1-hour, 6-hour and 24-hour) for the usage of Visualization GUI.		X			
1	1	3.5.2	CSA should provide an on-demand online visualisation capability (Visualization GUI), usage of Visualization GUI.		X			
1	1	3.5.3	It should be possible to download plot files corresponding to online plots, and the corresponding CEF files. These files should be available both at screen resolution and publishable quality in an efficient format such as jpeg or gif, and/or as a drawn vector file such as pdf or eps.		X			
1	1	3.5.4	The visualization GUI must allow to select any combination of plots from different Cluster spacecraft and instruments.		X			
2	1	3.5.5	The visualization GUI must allow to combine the data from several spacecraft or several instruments into the same panel (on the same y-axis range).		X			
2	1	3.5.6	A set of default plot styles must be supported, and these are to include multi-trace line plots, stack plots, status bars and spectrograms with control over colours and styles together with sufficient labelling.		X			
1	1	3.5.7	The user should be able to save the requested sets of graphical variables as graphical profile for using them in future.		X			
3	3	3.5.8	Zoom in and out should be supported on plots.		X			
1	1	3.5.9	Visualization tool must have next/previous buttons.		X			
1	1	3.5.10	Plotting interval can be selected either with start/stop time or with start time/duration.		X			
1	1	3.5.11	The user must be able to arrange the order of plot panels.		X			
Inventory Plots								
1	1	3.6.1	CSA should be able to create and display inventory plots.		X			
1	1	3.6.2	The inventory tool should provide next/previous buttons for the inventory plots.		X			
1	1	3.6.3	At the inventory tool the user should be able to view annual/monthly/daily inventory plots.		X			
3	2	3.6.4	The inventory tool should allow the user to create an inventory of any combination of datasets, up to 56 hours (= duration of one Cluster orbit).		X			
Quick-look Plots								
2	1	3.7.1	CSA should provide a browsing tool for the CSA quick-look plots (QL tool).		X			
2	1	3.7.2	in the QL browsing tool the user can select any number of consecutive plots and reduce the size of the plots on screen.		X			

11.2 Test/analysis/inspection/review of design to requirement traceability matrix

Note that here only the requirement ID is used:

Test Method	Requirements IDs
Test	[2.1.1], [2.1.2], [2.1.3], [2.1.4], [2.1.6], [2.1.7], [2.1.8], [2.1.9], [2.1.10], [2.2.1], [2.2.2], [2.2.3], [2.2.4], [2.2.5], [2.3.1], [2.4.1], [2.5.1], [2.5.2], [3.1.1], [3.1.2], [3.1.3], [3.1.4], [3.1.5], [3.1.6], [3.1.7], [3.1.8], [3.1.9], [3.1.10], [3.1.11], [3.1.12], [3.1.13], [3.1.14], [3.1.15], [3.1.16], [3.1.17], [3.1.18], [3.1.19], [3.1.20], [3.2.1], [3.2.2], [3.2.3], [3.2.4], [3.2.5], [3.2.6], [3.2.7], [3.3.1], [3.3.2], [3.3.3], [3.4.1], [3.4.2], [3.4.3], [3.5.1], [3.5.2], [3.5.3], [3.5.4], [3.5.5], [3.5.6], [3.5.7], [3.5.8], [3.5.9], [3.5.10], [3.5.11], [3.6.1], [3.6.2], [3.6.3], [3.6.4], [3.7.1], [3.7.2],
Analysis	[2.1.1], [2.1.2], [2.1.3], [2.1.4], [2.1.8], [2.1.9], [2.1.10], [2.2.1], [2.2.4], [3.1.8], [3.2.5], [3.3.3.]
Inspection	[2.1.5], [2.2.3], [2.5.1], [2.5.2], [3.1.2], [3.1.3], [3.1.6], [3.1.10], [3.1.15]
Review of design	[2.1.5]

11.3 Requirement to Test Design traceability matrix



SAT_CSA_URD_Test
s_Matrix_17Oct2013.